UNIVERSIDAD CARLOS III DE MADRID

Ingeniería Técnica en Informática de Gestión

Estudio comparativo de herramientas para la automatización de pruebas software

PROYECTO FIN DE CARRERA

Madrid, Octubre 2015

Autor: Daniel Álvaro Pérez

Tutor: Javier Fernández Muñoz

Título: Estudio comparativo de herramientas para la automatización de pruebas software.

Autor: Daniel Álvaro Pérez

Director: Javier Fernández Muñoz

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

A mis padres, por su ayuda y la paciencia que han tenido conmigo.

A mi hermano, por estar siempre ahí cuando lo he necesitado.

A mi otra mitad Almudena, por ser todo para mí y gracias a la cual he dado por fin este bonito paso.

A mi tutor Javier, que ha confiado en mí y le estaré siempre agradecido.

No hay manera conocida de agradeceros todo lo que habéis hecho por mí, por lo que simplemente os estaré eternamente agradecido.

"Creer posible algo es hacerlo cierto"
(Friedrich Hebbel)

Resumen

Este proyecto describe en primer lugar la importancia de la realización de pruebas de calidad dentro del ciclo de vida de un producto software y de cómo la automatización de estas pruebas provoca mejoras notables tanto para los usuarios como para los productos resultantes. El principal objetivo de este estudio es realizar una comparativa cuantitativa y cualitativa entre algunas de las herramientas gratuitas y herramientas de pago de automatización de pruebas software que existen en el mercado actualmente. En este documento se presentarán varias pruebas diseñadas, implementadas y ejecutadas en varias plataformas, para el posterior análisis de los resultados obtenidos y concluir qué tipo de herramienta es el óptimo.

Abstract

This project first describes the importance of quality testing in the life cycle of a product and how software automation of these tests leads to significant improvements for both users and the resulting products. The main objective of this study is to make a quantitative and qualitative comparison between some of the free tools and paid tools automation software tests on the market. This document designed several tests, implemented and executed on various platforms, for further analysis of the results and conclude what is the optimal tool will be presented.

Índice de contenidos

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y Objetivos

En este capítulo se presenta una visión general, resumida y estructurada de lo que se va a abordar en este proyecto. También se describirán las fases que se van a pasar para poder implementar finalmente las pruebas. Además, en este apartado se pondrá especial interés en mostrar los principales objetivos que se quieren obtener tras el estudio que se va a realizar.

Todos estos puntos serán expuestos en este capítulo a modo de introducción para ayudar al lector a entender y sintetizar de mejor manera todo el contenido.

1.1 Visión general

El motivo central de este trabajo es conocer cómo funciona la automatización de pruebas software y realizar una comparativa entre herramientas gratuitas y de pago.

Para llegar a esta valoración final, anteriormente se describirá la importancia de realizar pruebas software, su gran consideración dentro de la fase de calidad y en consecuencia del ciclo de vida de un aplicativo. También se explicará su evolución a las pruebas automatizadas y de cómo la utilización de varios programas existentes en el mercado nos permite alcanzar las metas deseadas.

La evaluación de las herramientas se llevará a cabo mediante la consecución de varias pruebas, las cuales se encontrarán diferenciadas claramente en funcionales y no funcionales.

Todos los casos de prueba serán analizados, diseñados y desarrollados para una serie de útiles libres y comerciales previamente seleccionados. Las pruebas serán ejecutadas dentro de unos escenarios deseados y se obtendrán resultados que serán posteriormente valorados para llegar a la conclusión final.

1.2 Objetivos

El objetivo principal de este proyecto es realizar un estudio y concluir con qué herramientas de automatización para pruebas de software son más óptimas, las gratuitas o las de pago. Analizando e interpretando cuales son las ventajas e inconvenientes de utilizar unas u otras, llegando finalmente a una conclusión clara y concisa al respecto.

Pero este objetivo principal no es más que una finalidad general que engloba otras metas también intencionadas:

- Comprender la importancia de realizar pruebas de software si queremos alcanzar un producto final que sea de calidad.
- Entender cuáles son los costes, tanto directos como indirectos, de implementar o no unas buenas costumbres de testing.
- Conocer cuáles son las diferentes pruebas funcionales y no funcionales (pruebas de estrés, carga, etc.) que existen.
- Ver las ventajas que nos aporta la automatización de pruebas tanto a nivel de usuario como a nivel global.
- Conocer las herramientas de automatización más relevantes.

Tras la conclusión final, se mostrarán unos anexos con los que se pretende acercar más alguna de las herramientas al lector y que pueda empezar a utilizarla como un usuario más, de una manera más familiar.

Estos documentos finales han sido diseñados y creados particularmente desde la experiencia personal alcanzada.

1.3 Fases de desarrollo

El desarrollo de este proyecto se puede dividir en 6 fases, claramente diferenciadas.

La primera fase consiste en el estudio de las diferentes herramientas automáticas y las tecnologías que se utilizan actualmente dentro del ámbito de la calidad de software. Una vez realizado esa investigación, se decide qué herramientas vamos a utilizar y el aprendizaje necesario de las mismas.

En el siguiente punto, se realiza un análisis más completo tanto de las herramientas que se van a utilizar como de las pruebas que se van a realizar, tratando los motivos de las mismas y la finalidad que buscamos con ellas.

Tras ello, se diseña el juego de casos de prueba planificados y de los escenarios de ejecución que albergarán las pruebas, las cuales incluyen propiedades como la carga, tiempos de espera, franjas horarias de ejecución, número de iteraciones etc.). Todo esto se llevará a su implementación y se crean los scripts automatizados mediante las diferentes herramientas y se implantan.

En este paso también se ejecutan pruebas manuales, ejecutan pruebas de carga y pruebas funcionales automatizadas y se anota su comportamiento.

Tras ello, se analizar los resultados obtenidos y realizar una comparativa entre las herramientas utilizadas.

Finalmente, se confecciona y se redacta este documento.

1.4 Estructura del documento

A continuación, se explica de manera concisa los puntos que se abordan en cada uno de los apartados del documento.

Capítulo 1.

En esta parte se resumen tanto lo que se va a ver en el interior del documento, como los objetivos que se pretenden alcanzar al finalizar el mismo. Esta introducción tiene como objetivo ayudar con la percepción del contenido del proyecto.

Capítulo 2.

Nos acerca la importancia de las pruebas de software, su evolución hasta la utilización de las herramientas de automatización, los diferentes tipos que existen y cuáles son las más significativas en el área de las pruebas automáticas.

Capítulo 3.

En este capítulo se realiza el análisis de los requisitos y se determina qué pruebas se desean realizar, con qué herramientas y en qué plataformas y aplicaciones.

Capítulo 4.

Se diseñan e implementan varias pruebas con las que alcanzar los objetivos pretendidos. También se ejecutan los scripts en los escenarios implantados.

Capítulo 5.

En este apartado se plantea el objetivo central del proyecto. En primer lugar, se analizan y evalúan los resultados obtenidos en las pruebas realizadas en el anterior punto. Y posteriormente, se realiza una comparativa entre las herramientas gratuitas y de pago utilizadas. Dentro de esta sección se incluyen una serie de test realizados a un grupo de testers usuarios potenciales de herramientas de este tipo.

Capítulo 6.

Desglosa la planificación a seguir y el resumen de los costes para llevar a cabo este estudio.

Capítulo 7.

Se desarrollan las conclusiones que se deducen del estudio realizado. Así como, valoraciones propias al respecto.

Capítulo 2

Estado de la cuestión

En este capítulo se realiza un exposición de cómo la fase de pruebas de software siempre ha ido muy unida a la fase de desarrollo. Y de cómo estas pruebas han evolucionado a lo largo del tiempo convirtiéndose en parte importantísima y necesaria dentro del proceso para el desarrollo del producto software y de las empresas encargadas de los mismos.

También entraremos en una descripción resumida de las herramientas que aparecen en el mercado actual y que tienen potencial para ser utilizadas como software de automatización.

2.1 Evolución de las pruebas de calidad de software.

2.1.1 Historia e importancia de la calidad de software.

La calidad del software es una preocupación a la que se dedican cada vez mayor número de recursos. Todo proyecto informático tiene como objetivo producir software de la mejor calidad posible, que cumpla, y si puede supere las expectativas de los usuarios.

La calidad del producto, junto con la calidad del proceso, son algunos de los aspectos más importantes actualmente en el desarrollo de Software.

Aunque la calidad del software es un concepto relativamente reciente, realmente la calidad y concretamente las pruebas de software existen desde que existe el desarrollo de aplicaciones. Por ello, son muchos los esfuerzos desarrollados en este campo en las últimas décadas, consiguiendo grandes avances y mejorando la convergencia con la calidad tradicional.

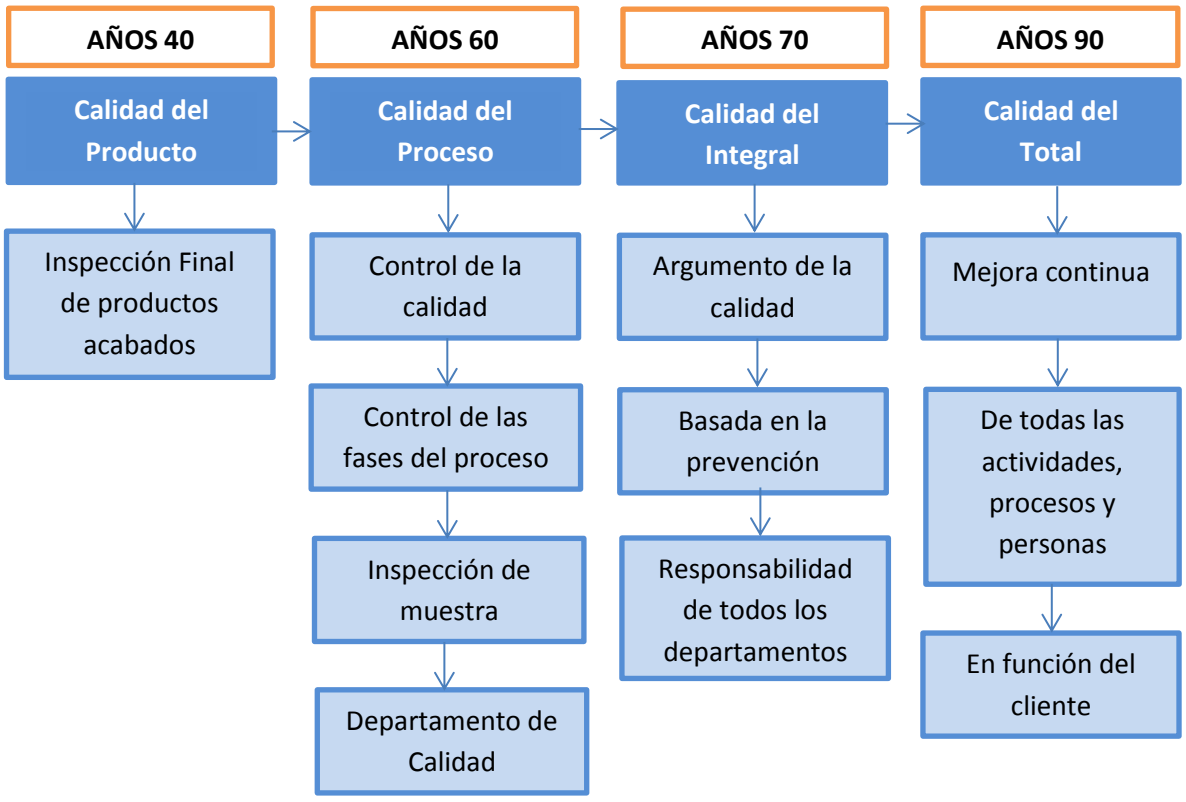


Figura 1 - La calidad a lo largo de la historia.

En el mundo del software, y supongo que en otras disciplinas también, cuando nos referimos al amplio concepto de "calidad software" hemos de ser muy conscientes de que en realidad ese concepto de calidad se subdivide, principalmente, en tres tipos de calidad: la del proceso, la del producto y la de las personas/equipos.

Figura 2 - Dimensiones de la calidad software

Nosotros nos centraremos en la calidad del producto - proceso, puesto que es esta la calidad que podemos evaluar mediante las herramientas de automatización.

La calidad vista desde el mundo de los procesos nos dice que la calidad del producto software está determinada por la calidad del proceso. Por proceso se entienden las actividades, tareas, entrada, salida, procedimientos, etc., para desarrollar y mantener software.

Para una empresa que no desarrolla, que adquiere productos software desarrollados por terceros (externalización), la certificación de la calidad del proceso de su subcontratista puede ser condición necesaria e importante como garantía de calidad, sobre todo en procesos de selección de proveedores, cuando aún no está el software desarrollado, pero puede no ser suficiente para garantizar la calidad del producto.

Será la calidad del producto la que evidenciará inequívocamente la calidad del mismo, sin necesidad de suposiciones; un conjunto coherente de métricas e indicadores del producto estructurados según un modelo tipo.

Si una empresa que desarrolla software debe preocuparse de la calidad del proceso y del producto que desarrolla y entrega, una empresa que sólo compra software (el típico cliente) debería, principalmente, preocuparse de la calidad del producto que compra.

Está claro por lo tanto que la inversión en calidad, sobre todo si ésta comienza desde las primeras fases de desarrollo, provocará la creación de un producto más robusto y de mayor confianza. Esto suscitará un menor número de problemas cuando ya se encuentra en el entorno de producción y en consecuencia, un menor gasto en mantenimientos y resolución de problemas del software comercializado.

Tras corroborar que la calidad del producto es óptima se debe de comprobar que el software "funcione" y esto es lo que vamos a evaluar gracias al testing.

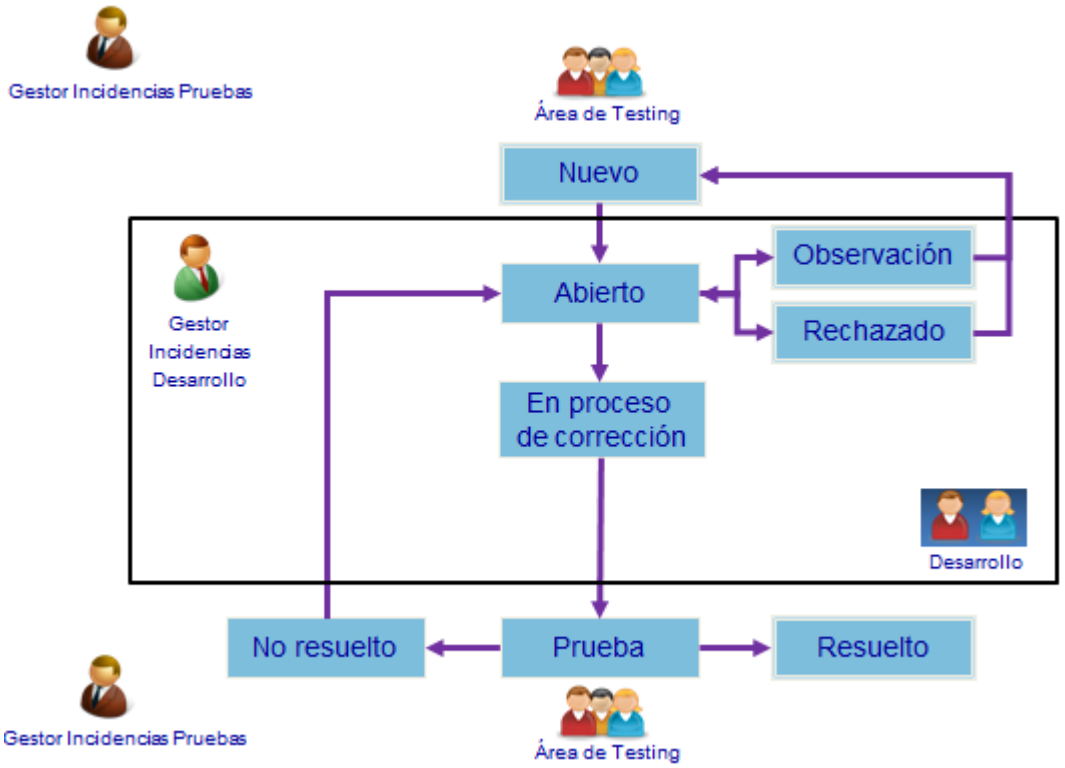


Figura 3 - Ciclo de vida software

El testing se define como una investigación técnica de un producto bajo prueba con el fin de brindar información relativa a la calidad del software, a los diferentes actores involucrados en un proyecto.

A partir de la información obtenida del testing se pueden tomar decisiones. Las decisiones pueden ser desde cuándo liberar un producto a producción, conociendo los riesgos que esto implica, hasta cómo mejorar las diferentes áreas dentro de la empresa. En definitiva, el testing es un agente de cambio, lo importante es interpretar la información obtenida para que todos los actores puedan actuar en forma oportuna donde sea necesario.

En el software la confianza es un elemento importante ya que ciertos fallos pueden tener consecuencias indeseables como pérdidas de dinero, negocios e incluso de vidas, dependiendo de qué tan crítico sea el ámbito en el que el software interactúa. Las pruebas le dan un valor añadido a cada proyecto brindando confianza a los distintos actores.

Figura 4 - Calidad pieza importante

2.1.2 Definición y tipos de pruebas software.

Vamos a realizar la clasificación de las pruebas software en función del objetivo en que se centran. Este objetivo puede ser cualquiera de los siguientes:

- Una función a realizar por el software
- Una característica de calidad no funcional, tales como la fiabilidad o la usabilidad.
- La estructura o arquitectura del software o sistema.

Estas pruebas confirmarán que se han solucionado los defectos (pruebas de confirmación) y también localizarán cambios no intencionados (pruebas de regresión).

Niveles de pruebas o testing				
Test	**Objetivo**	**Participantes**	**Ambiente**	**Método**
Unitario	Detectar errores en los datos, lógica, algoritmos	Programadores	Desarrollo	Caja Blanca
Integración	Detectar errores de interfaces y relaciones entre componentes	Programadores	Desarrollo	Caja Blanca, Top Down, Bottom Up
Funcional	Detectar errores en la implementación de requisitos	Testers, Analistas	Desarrollo	Funcional
Sistema	Detectar fallos en la cobertura de los requisitos	Testers, Analistas	Desarrollo	Funcional
Aceptación	Detectar fallos en la implementación del sistema	Testers, Analistas, Cliente	Productivo	Funcional

Tabla 1 - Pruebas o Testing (Tipos)

Las **pruebas funcionales** se basan en el funcionamiento del producto, buscando si la solución satisface las necesidades por la que fue creada. Este tipo de testing permite determinar si se ha construido el software deseado y si es oportuno liberar la versión del producto al mercado.

Este análisis también permite a las empresas, que hacen uso intensivo de las tecnologías de la información, determinar si han adquirido el software deseado.

Para ello, se propone una estrategia basada en el análisis de riesgos del producto, definiendo claramente el contexto y los objetivos. Y se elabora y planifican una serie de pruebas basadas en las funcionalidades prioritarias, considerando su complejidad y criticidad.

Estas pruebas tienen en cuenta el comportamiento externo del software (pruebas de caja negra). Es decir, se centrará en probar una serie de navegaciones, normalmente detalladas por los desarrolladores, para conseguir un resultado final deseado, sin preocuparse de lo que ocurre en el interior.

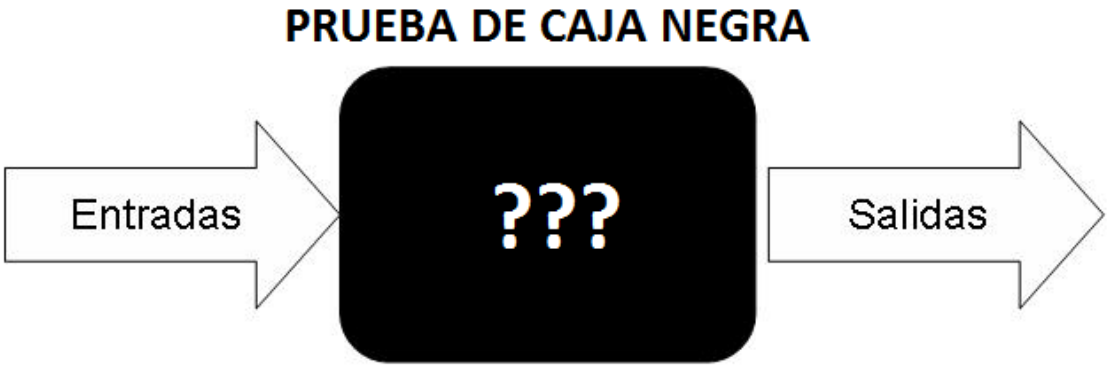


Figura 5 - Pruebas de caja negra

Las **pruebas no funcionales** son complementarias a las anteriores y se centran en pruebas necesarias para medir las características de los sistemas software, es decir tienen por objetivo evaluar "cómo" funciona el producto software. Este tipo de testing puede cuantificarse mediante diferentes escalas según el análisis realizado (tiempos de respuesta, operaciones por segundo etc.)

Esta categoría de pruebas tienen en cuenta el comportamiento externo del software y, en la mayoría de los casos, para ello utiliza técnicas de diseño de pruebas de caja negra también.

A esta clase de test se corresponden, entre otras, las pruebas de rendimiento, pruebas de carga, pruebas de estrés, pruebas de mantenibilidad , pruebas de fiabilidad y pruebas de portabilidad.

Las **pruebas estructurales** son una aproximación al diseño de casos de prueba en donde las pruebas se derivan a partir del conocimiento de la estructura e implementación del software.

Esta aproximación de puede denominar de caja blanca o de caja de cristal, ya que se centra en los detalles procedimentales del software, por lo que su diseño está estrechamente ligado al código fuente.

Estas pruebas de arquitectura pueden realizarse en todos los niveles de prueba, pero especialmente en las pruebas de componente y en las pruebas de integración, y consisten en la evaluación de la cobertura de un tipo de estructura.

La cobertura es la medida en que un juego de pruebas ha probado una estructura, expresada como porcentaje de los elementos cubiertos.

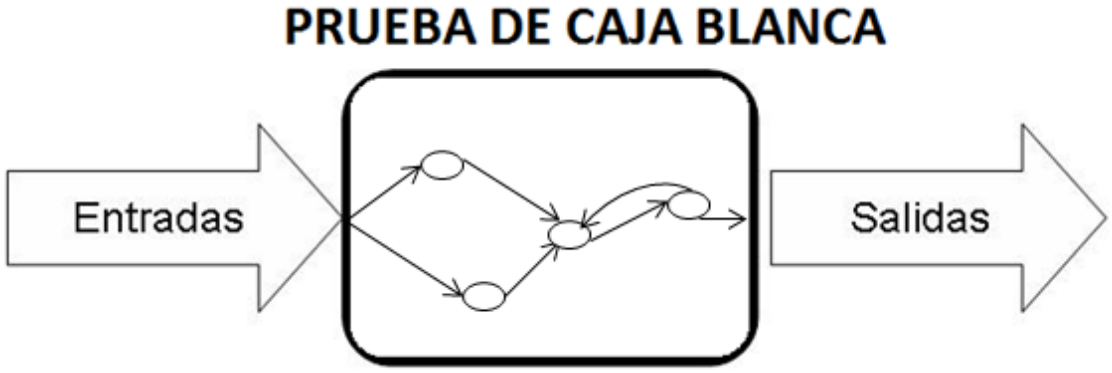


Figura 6 - Pruebas de caja blanca

Las pruebas asociadas a cambios, que consisten en repetición de pruebas y **pruebas de regresión**. Estas pruebas aparecen tras detectarse y corregirse un defecto, ya que el software tiene que volver a probarse para realizar la confirmación de que el defecto original se ha arreglado de manera satisfactoria.

Las pruebas de regresión consisten en la prueba reiterada de un programa ya probado, que tras haber sido modificado, pretende localizar defectos surgidos como resultado del cambio o cambios que se hayan realizado. El alcance de estas pruebas depende de no encontrar defectos en el software que antes funcionaba correctamente.

Estas pruebas deben de ser repetibles y contener al menos, las funcionalidades principales y más críticas del producto software. Las pruebas de regresión se pueden ejecutar en todos los niveles de prueba, e incluyen pruebas funcionales, no funcionales y estructurales.

Los juegos de prueba de regresión se ejecutan muchas veces y por lo general son de lenta evolución, por lo que las pruebas de regresión constituyen un gran potencial para la automatización.

2.2 Software libre contra Software comercial.

2.2.1 Características del SW gratuito y SW comercial.

En primer lugar, se debe diferenciar entre software libre (Software Open Source) y software gratuito (Free Software), ya que son términos que normalmente vienen estrechamente ligados y para los que compartir es una de las bases principales. Sin embargo, no se deben de confundir.

El **Software Libre**, es el que es desarrollado y distribuido libremente y que además de compartir su los aplicativos utilitarios, también comparte el código fuente.

Figura 7 - Software Libre

Este software respeta las cuatro libertades que la FSF (Free Software Foundation) establece:

- La libertad de usar el programa, con cualquier propósito.
- La libertad de estudiar cómo funciona el programa y modificarlo, adaptándolo a tus necesidades.
- La libertad de distribuir copias del programa, con lo cual puedes ayudar a tu prójimo.
- La libertad de mejorar el programa y hacer públicas esas mejoras a los demás, de modo que toda la comunidad se beneficie.

En resumen, es aquel que respeta la libertad de todos los usuarios que adquirieron el producto para ser usado, copiado, estudiado, modificado, y redistribuido libremente de varias formas. Es muy importante aclarar que el Software Libre establece muchas libertades pero no es necesariamente gratuito.

El **Software gratuito**, es el que podemos adquirir y utilizar sin necesidad de pagar dinero a cambio de una licencia que nos permita trabajar con este software. Por regla general, son aplicaciones con funciones específicas y que podemos catalogar en diferentes tipos.

- Versiones de condicionado: Aplicaciones como MSN Messenger, Yahoo Messenger, Picasa, Vimeo Uploader y aquellas ligadas a una empresa o servicio web son herramientas de uso libre.

Figura 8 - Software versiones condicionado

- Versiones de prueba: son las que ofrecen software con unas funcionalidades básicas (demo) o con tiempos limitados (shareware). Si necesitásemos usar más bloques funcionales o utilizarlo por más tiempo nos proponen una versión PRO, generalmente con un coste ligado al mismo.

- Pago por servicios: un ejemplo claro de este tipo es SKYPE, ya que este software que nos permite hacer llamadas de PC a PC de forma gratuita sin límite, pero de PC a fijos y móviles conlleva un pago.

Figura 9 - Software pago por servicios

- Distribución gratuita: Son aplicaciones que no están ligadas a ningún tipo de pago por licencias.

En contraposición al software anteriormente expuesto, se encuentra el **software de pago**, propietario o privativo. Es el más conocido y consiste básicamente en pagar por un programa para poder instalarlo en nuestro ordenador y utilizarlo.

Con el pago de estas licencias el usuario tendrá, por regla general, una garantía de que el programa funcionará de manera correcta y en caso de no hacerlo, tener el derecho a un servicio de asistencia técnica que ayude a solventar el problema.

Este tipo de herramientas software no son libres o lo son sólo parcialmente (semilibre), ya que el usuario tiene limitaciones para usarlo o modificarlo, y tienen totalmente prohibida su copia y distribución.

Figura 10 - Software privativo

2.2.2 Comparativa general.

Las principales diferencias entre el software privativo y el software libre gratuito se resumen en la siguiente tabla.

	Software Privativo	Software Libre
Adquisición de licencias	SI	NO
Modificación del Código Fuente	NO	SI
Posibilidad de hacer y distribuir copias	NO	SI
Marketing y publicidad del SW	ALTO	BAJO
Formación existente para utilizar el software	ALTA	BAJA
Garantía de funcionamiento	SI	NO
Calidad y amigabilidad interfaz gráfica	ALTA	BAJA
Compatibilidad con hardware	ALTA	MEDIA
Alcance funcional y operativo del software	ALTO	BAJO – MEDIO
Coste de construcción la aplicación software	ALTO	BAJO
Soporte disponible para la aplicación	SI	NO
Seguridad y anti-virus	MEDIA	ALTA
Requisitos Hardware	BAJO	MEDIO

Tabla 2 - Comparativa entre Software privativo y software libre

El software libre a pesar de que no implica el que sea gratuito, lo habitual es que podamos obtener los programas descargándolos libremente de Internet o a través de distribuciones. El software propietario conlleva pagar por adquirir estos programas.

2.3 Pruebas automatizadas.

2.3.1 En qué consiste la automatización de pruebas.

La automatización de pruebas es un tipo particular de sistema software y consiste en la realización de pruebas basadas en herramientas que permiten su ejecución y análisis automático. Se trata de una alternativa muy interesante cuando se quiere asegurar cierto nivel de calidad de cada liberación de productos o versiones software que se realice.

Lo ideal para empezar a utilizarla, es que nos encontremos en una fase de cierta estabilidad dentro de los cambios y de las funcionalidades básicas que tiene el producto (regresión). Ya que si el software cambiase bastante en unos intervalos cortos de tiempo, provocaría mucho coste en el mantenimiento de scripts automáticos y no sería aplicable.

Esta automatización es muy útil sobre todo, cuando al aumentar la productividad en el desarrollo software, disminuyen los tiempos entre la generación de nuevas versiones y no queremos que crezca la incertidumbre acerca de la calidad del producto software resultante.

Utilizando este método se pretende que el número de errores detectados y reportados en el entorno de producción sean mínimos. Esto es muy importante, ya que según antes se detecte un error en el ciclo de vida del software menor serán el coste de haberlo encontrado y reparado, además de los daños de imagen del producto que se provocarían si estos fallos se encontrasen en producción.

El objetivo de la automatización de pruebas no es eliminar las pruebas manuales por completo, ni suplantar a los testers manuales. El objetivo del proceso de automatización es apoyar y ser un añadido, aportando:

- Apoyo en pruebas de regresión frecuentes.
- Iteraciones ilimitadas de ejecución de casos de prueba.
- Mayor flexibilidad y potencial de la ejecución de pruebas.
- Reporte personalizado de los defectos de la aplicación.
- Apoyo a las metodologías de desarrollo ágil.
- Documentación normalizada de casos de prueba.

2.3.2 Tipos de pruebas automatizadas.

Gestión de pruebas

- Automatización de pruebas **funcionales**: Estas requieren una interfaz gráfica en la que se llevarán a cabo las navegaciones pertinentes.

  Se trata de realizar scripts, normalmente basados en un lenguaje de programación, mediante los cuales se podrá reproducir la funcionalidad deseada sobre la interfaz.

  Estos scripts se generar mediante el método de grabar y reproducir (Record and Playback), programando directamente los scripts en algún lenguaje de programación (scripting), etc.

- Automatización enfocada a estudiar características de calidad **no funcionales**, tales como la eficiencia, fiabilidad o la usabilidad.

  En este tipo de pruebas se encuentran las pruebas de rendimiento, pruebas de prestaciones, de seguridad, portabilidad etc.:

  - Prueba de carga: Tipo de prueba relacionado con la medida del comportamiento de un componente o sistema con una carga creciente, por ejemplo el número de usuarios concurrentes y/o número de transacciones para determinar qué carga puede ser soportada por el componente o sistema.

Figura 11 - Gráficos flechas

- Prueba de rendimiento: Se centra en determinar la velocidad con la que el sistema bajo pruebas realiza una tarea en las condiciones particulares del escenario de pruebas. Este servicio ayuda a su organización a detectar los cuellos de botella de su aplicación, antes de que sus usuarios sufran un mal rendimiento, con la consecuente pérdida económica y frustración de sus clientes o empleados.

- Prueba de estrés: Pruebas orientadas a evaluar un componente o sistema en, o más allá, de los límites especificados en los requisitos.

Gracias a la automatización de este tipo de pruebas se permite alcanzar cotas y realizar algunos tipos de pruebas que no se podrían obtener de manera manual. Esto se debe al potencial que nos ofrecen las herramientas encargadas de este tipo de pruebas y de la infraestructura hardware que utilicemos.

2.3.3 Ventajas de automatizar pruebas.

La automatización de pruebas obtiene grandes resultados tras realizar una inversión a medio plazo, ya que existe un tiempo de aprendizaje de las unidades de las herramientas y de la generación de los scripts.

Figura 12 - Engranajes Automatización

En un principio, invertir en automatización se traduce en mayores costes fijos provenientes de la adquisición de hardware, licencias, formación… Pero después de un tiempo asumible se obtendrán una serie de ventajas evidentes:

- Mejorar la eficiencia de las pruebas automatizando tareas repetitivas o dando soporte a las actividades de pruebas manuales, como la planificación, el diseño, la elaboración de informes y la monitorización de pruebas.

- Automatizar aquellas actividades que requieren muchos recursos si se hacen de forma manual (por ejemplo, pruebas estáticas).

- Reasignación de recursos que previamente realizaban las pruebas de manera manual, y utilizarlos para otra tarea dentro del negocio.

- Automatizar aquellas actividades que no pueden ejecutarse de forma manual (por ejemplo, pruebas de rendimiento a gran escala de aplicaciones cliente - servidor).

- Aumentar la fiabilidad de las pruebas (por ejemplo, cuando se automatizan comparaciones de grandes ficheros de datos y se simula su comportamiento).

- Posibilidad de programar la ejecución de un plan de pruebas o escenario deseado, de manera desatendida, del modo que se desee.

- Posibilidad de realizar algunos tipos de prueba en menos tiempo.

2.4 Tecnología disponible en el mercado actual.

2.4.1 Arquitectura de las herramientas de automatización.

Hay varias herramientas que dan soporte a distintos aspectos de las pruebas. Sin embargo, el software utilizado para automatizar se constituye, generalmente, de una serie de componentes:

- Editor: Es el módulo donde se van a crear y modificar los casos de prueba.
- Caso de prueba (Test Case o Test Set): Es el repositorio de los casos de prueba, con la estructura y lenguaje propios de cada uno de ellos.
- Intérprete (Interpreter): Traductor del lenguaje de los casos de prueba.
- Adaptador (Adapter): Son las capas de adaptación entre los casos de prueba interpretados y el sistema software que se está probando.
- Comparador (Comparator): Es el mecanismo que permite comprobar los resultados esperados y los obtenidos en realidad.
- Ejecutor (Runner): Permite correr los casos de prueba. Dependiendo de la herramienta y las pruebas puede permitir ejecutar secuencial, en paralelo, etc.
- Generador de resultados (Reporter): Facilita la creación de diferentes tipos de informes de resultados de las pruebas.

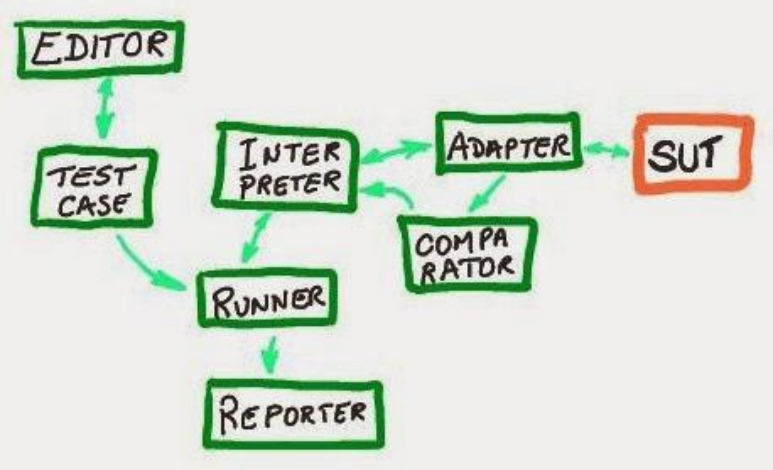


Figura 13 - Arquitectura herramienta automatización.

2.4.2 Herramientas libres.

Herramientas para gestión de pruebas

- **TestLink**: es una herramienta que permite crear y gestionar casos de pruebas y organizarlos en planes de prueba. Estos planes permiten a los miembros del equipo ejecutar casos de test y registrar los resultados dinámicamente, generar informes, mantener la trazabilidad con los requerimientos, así como priorizar y asignar tareas.

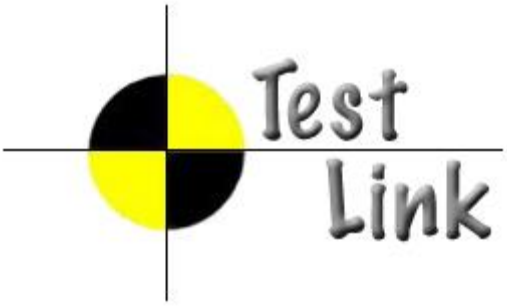


Figura 14 - Logo TestLink

- **Bugzilla Testopia:** es un administrador de casos de prueba, el cual maneja extensiones para interactuar con Bugzilla. Testopia está diseñado para ser una herramienta genérica para el seguimiento de casos de prueba, permitiendo a las organizaciones realizar las pruebas de software e integrar reportes de defectos encontrados, así como el resultado de los de los caso de prueba. Testopia está diseñado desde el punto de vista de la actividad de pruebas, este puede ser usado para llevar el seguimiento de pruebas, así como el seguimiento virtual de cualquier proceso de ingeniería.

Herramientas para pruebas de carga y rendimiento

- **Apache JMeter:** Es el software gratuito y de código abierto para pruebas de prestaciones por excelencia. Una aplicación 100 % Java diseñado para realizar pruebas de carga y medir el rendimiento. Fue diseñado originalmente para pruebas de aplicaciones web, pero desde entonces se ha expandido a otras funciones de prueba. Puede ser utilizado para probar el rendimiento tanto en recursos estáticos y dinámicos (Webservices (SOAP / REST) , lenguajes dinámicos Web - PHP , Java , ASP.NET , archivos , etc. - , objetos Java , bases de datos y consultas , Servidores FTP y mas). Se puede utilizar para simular una carga pesada en un servidor, en un grupo de servidores, en la red o en el objeto software para probar su resistencia o para analizar el rendimiento general bajo diferentes tipos de carga. Se puede utilizar para hacer un análisis gráfico de rendimiento o para probar su comportamiento / objeto de servidor / script bajo carga concurrente pesada.

Figura 15 - Logo JMeter

- **FWPTT load testing:** se trata de una herramienta enfocada a realizar pruebas de carga en aplicaciones web. Permite grabar peticiones normales y ajax. Probado para aplicaciones ASP, .NET, JSP, PHP u otro. Puede grabar sus acciones de navegación utilizando cualquier navegador IE, Firefox etc.

Figura 16 - Logo fwptt

Herramientas para automatizar pruebas funcionales

- **Selenium**: Se trata de la herramienta gratuita más utilizada para automatizar pruebas funcionales. Este software ofrece la ventaja de ser una herramienta de código abierto. Esta herramienta está diseñada exclusivamente para navegadores, es decir, que está diseñada para tecnologías web. Se puede escribir los scripts tanto en C#, Java, Groovy, Perl, PHP, Python y Ruby. En cuanto al soporte, viene dado a través de la comunidad que lo desarrolla y colabora en ello, pero eso no garantiza una respuesta. Esta herramienta se puede usar en Windows, Linux y MacOS.

Figura 17 - Logo Selenium

- **Watir:** Es una familia de librerías Ruby de Código Abierto (Open Source) para la automatización de navegadores web.
  Le permite a su usuario escribir pruebas fáciles de leer y mantener. Sencilla y flexible. Tiene la capacidad de hacer clic en enlaces, llenar formularios de pantallas con datos y presionar botones.
  Watir también revisa los resultados, incluyendo verificar si los textos esperados se muestran en las páginas. Tiene la capacidad de enlazarse con bases de datos, leer archivos de datos y hojas de cálculo, exportar XML y estructurar los códigos como librerías reutilizables.

Figura 18 - Logo Watir

- **SoapUI:** Es una solución multiplataforma para pruebas funcionales. Con una interfaz gráfica amigable, SoapUI permite crear y ejecutar pruebas funcionales, de regresión y de carga automatizadas con facilidad y rapidez. En un solo entorno de prueba, SoapUI ofrece cobertura de la prueba completa y apoya todos los protocolos y tecnologías estándar.

Figura 19 - Logo SoapUI

2.4.3 Herramientas de pago.

Herramientas para gestión de pruebas

- **HP ALM** (HP Application Lifecycle Management): Anteriormente conocido como Quality Center. Se trata de una solución de gestión de aplicaciones para ayudar a definir, crear, probar y entregar aplicaciones de forma rápida y con confianza en todo el ciclo de vida de desarrollo del software.
  Es la herramienta líder para la Gestión de Calidad (requisitos, pruebas y defectos) en proyectos de desarrollo de aplicaciones en los departamentos de IT.

Figura 20 - Logo HP ALM

Herramientas para pruebas de carga y rendimiento

- **HP LoadRunner:** Es un software de pruebas de carga que ofrece una perspectiva precisa del rendimiento del sistema al completo para identificar y resolver problemas antes de lanzar la aplicación. Se trata de una herramienta única de pruebas de carga para aplicaciones móviles, web y heredadas que proporciona una visión precisa del rendimiento integral del sistema. Lo cual, permite identificar y corregir cualquier incidencia antes de salir al mercado.

Figura 21 - Logo HP LoadRunner

- **IBM Rational Performance Tester:** Es un motor de pruebas de carga de trabajo que le permite validar la escalabilidad y el rendimiento de las aplicaciones web y de servidor. La solución maximiza el uso de la infraestructura de pruebas para desplegar rápidamente escenarios de carga y generar actividad de pruebas del sistema a gran escala.
  Esta herramienta permite realizar pruebas de rendimiento de las aplicaciones y resolver los problemas antes de que resulten costosos.

Figura 22 - Logo IBM Performance Tester

Herramientas para automatizar pruebas funcionales

- **HP Unified Functional Testing**: Antes conocida como Quick Test Professional (QTP). Soporta una variedad muy extensa de tecnologías: Java, Sap, Siebel, Visual Basic .Net y Oracle entre muchas otras. Esta Herramienta se basa en el reconocimiento de objetos, aunque se puede utilizar el posicionamiento dentro de una pantalla para la realización de pruebas, así como reconocimiento de texto por OCR. Al poseer un interfaz amigable y un código de generación de scripts en visual basic se consigue que se aprenda más fácilmente, obteniendo una curva de aprendizaje muy alta desde el primer momento. Esta herramienta se suele integrar con todas las herramientas de la suite de HP como Quality Center (gestor de requisitos, casos de prueba y defectos). Esta herramienta se instala en cualquiera de las versiones de Windows recientes.

Figura 23 - Logo HP UFT

- **Microsoft Test Manager (MTM):** Herramienta propiedad de Microsoft para la gestión y automatización de pruebas. Esta herramienta está incluida en Microsoft Visual Studio Ultimate 2010 o en Visual Studio Test Professional 2010. El interfaz y el código generado en los scripts es bastante intuitivo, se debe de integrar con Team Foundation Server que almacena los casos de prueba y requerimientos entre otras cosas. El código generado se llama coded UI que graba operaciones de interfaz basado en Visual C#.NET. Se instala en sistemas operativos Windows.

Figura 24 - Logo MTM

- **IBM Rational Functional Tester:** Herramienta de automatización de pruebas funcionales y de regresión. Proporciona capacidades de pruebas de interfaz gráfica, pruebas manejadas por datos (Data Driven), pruebas funcionales y pruebas de regresión.
  Algunas de sus características son: Simplificación de creación y visualización de pruebas, pruebas de tipo storyboards, trazabilidad en todo el ciclo de vida, validación de data dinámica (por medio de un wizard), e inclusive capacidad de definir scripts (por medio de lenguajes de Scripting).
  Rational Function Tester da soporte a diversas aplicaciones, como aplicaciones basadas en web, .Net, Java, Siebel, SAP, basadas en emulador de terminal, PowerBuilder, Ajax, Adobe Flex, Dojo Toolkit, GEF, documentos Adobe PDF, zSeries, iSeries y pSeries.

Figura 25 - Logo IBM Rational

- **Eggplant**: Es la solución que presenta la compañía Testplant. Esta herramienta es independiente de las tecnologías ya que utiliza la pantalla como imagen y mediante reconocimiento OCR es capaz de identificar imágenes y texto para su utilización. Tiene un interfaz sencillo aunque utiliza un código de generación de scripts muy poco extendido en la actualidad, Sense Talk, originalmente desarrollado por Next Step. Esta herramienta se puede integrar con otras muchas como Eggplant Manager, también de este fabricante. Se puede usar tanto Windows, MacOS y Linux.

Figura 26 - Logo eggPlant

- **Ranorex:** Se basa en reconocimiento de objetos y genera scripts tanto en C# como en Visual Basic. Además tiene un interfaz muy amigable. Se integra sólo con las herramientas propietarias de Ranorex para la gestión de casos de prueba.

Figura 27 - Logo Ranorex

- **TestComplete:** pertenece a SmartBear software y se trata de una herramienta orientada a objetos que soporta una gran cantidad de tecnologías tales como Visual Basic, Delphi, C + + y otras herramientas de desarrollo. Se puede ejecutar en los navegadores Internet Explorer, Mozilla Firefox y Google Chrome en sus versiones de 32 y 64 bits y soporta flash y otros complementos.

Figura 28 - Logo TestComplete

Capítulo 3

Tecnología empleada, pruebas a realizar y análisis del estudio.

En este capítulo se describirá de una forma más detallada las herramientas seleccionadas para formar parte de las pruebas que se van a realizar en este proyecto.

También se explicarán los motivos para implementar las pruebas que se han elegido, los objetivos de las mismas, las aplicaciones y arquitecturas que van a ser testeadas y la estrategia que se llevará a cabo para su elaboración.

3.1 Tecnología utilizada.

3.1.1 Selección de herramientas.

Las herramientas seleccionadas para el estudio y comparativa que se va a realizar son las que tienen un software con más potencial y que son punteras en el ámbito actual del testing automático de pruebas.

Estas herramientas, tanto comerciales como gratuitas, se caracterizan por ser las más utilizadas y las que aportan un mayor número de posibilidades técnicas en su ámbito.

Dentro del ámbito de la automatización de pruebas funcionales, el software gratuito más utilizado es Selenium, mientras que dentro de las de pago HP UFT es el líder.

Software Gratuito	Software Comercial
Selenium IDE	HP UFT

Tabla 3 - Herramientas automatización pruebas funcionales

Para las herramientas orientadas a pruebas de carga y rendimiento, la número uno en el sector de las gratuitas es JMeter y en las privativas HP LoadRunner.

Software Gratuito	Software Comercial
Apache Jmeter	HP LoadRunner

Tabla 4 - Herramientas automatización pruebas carga y rendimiento

Cabe recalcar, que las herramientas seleccionadas son las más completas y útiles dentro del tipo de pruebas al que están enfocadas. Y en consecuencia, brindan mayor cantidad de variantes tanto para la grabación como para la ejecución y recogida de los resultados obtenidos, que el resto de las herramientas. Lo que permite hacer un análisis más exhaustivo de las pruebas realizadas.

3.1.2 Detalle y fundamentos de la tecnología empleada.

A continuación, se va a realizar un pequeño resumen de los detalles, características y soluciones que nos aporta cada una de las herramientas indicadas:

- **Selenium:**

Se trata de la herramienta gratuita por excelencia para automatizar los navegadores Web. Para ello nos permite grabar, editar y depurar casos de prueba, que podrán ser ejecutados de forma automática e iterativa posteriormente. Principalmente se utiliza para automatizar las pruebas de aplicaciones Web, pero se lo puede usar para otras actividades, por ejemplo realizar parametrizaciones de sistemas web que podrían resultar aburridas al realizarlas manualmente.

Entre sus principales características podemos remarcar:

- Facilidad de grabación y ejecución de los test.
- Referencia a objetos DOM en base al ID, nombre o a través de XPath.
- Auto-completado para todos los comandos comunes.
- Las acciones pueden ser ejecutadas paso a paso.
- Herramienta de depuración y puntos de ruptura (breakpoints).
- Los test pueden ser almacenados en diferentes formatos.

Existen 2 formas de utilizar Selenium dependiendo de las necesidades o las características de la prueba que se desee realizar, “Selenium IDE” y “SeleniumWebDriver”.

Selenium funciona sobre cualquier Navegador (a excepción de Selenium IDE que funciona solo sobre Firefox) y sobre cualquier sistema operativo.

Selenium IDE

Permite crear Scripts de pruebas de manera rápida ya que trabaja como una “Grabadora” que captura todas las acciones que se hacen sobre la interfaz de usuario (una especie de macro),generando automáticamente los comandos del Script de pruebas. Es de gran ayuda en las pruebas de Regresión y Retesting.

Esta herramienta trabaja como un "complemento" (add-in) de Firefox, el cual permite Grabar y Reproducir de manera simple las iteraciones con el navegador.

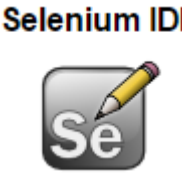


Figura 29 - Logo Selenium IDE

Otras Características a resaltar son:

- Manejo inteligente de la identificación de Campos o Elementos en la página Web, se puede usar IDs, nombres o XPath.
- Autocompleta todos los comandos de Selenium Debug y Puntos de quiebre.
- Permite grabar los Scripts en formato HTML, Ruby, JUnit, entre otras.
- Soporte para el archivo "user-extensions.js", el cual permite agregar funciones javascript personalizadas.
- Opciones para realizar validaciones (asserts) automáticas del título para cada página en la que se esté navegando.

Selenium WebDriver

Se trata de un API orientado a las aplicaciones web modernas, donde anteriormente había problemas para el testing de las mismas. Además tienes pasarelas de comunicación que funcionan directamente con el servidor en diversos lenguajes.

Figura 30 - Logo Selenium WebDriver

Selenium WebDriver permite crear Scripts de pruebas más robustos pues se puede desarrollar sobre varios lenguajes de Programación, entre ellos C# y Java. Es de gran ayuda en las pruebas de Regresión y Retesting.

Este componente permite además de la ejecución local, la ejecución remota haciendo uso de "Selenium Server".

- **HP UFT:**

HP Unified Functional Testing, propiedad de HP y anteriormente llamado Quick Test Professional (QTP), es la herramienta líder para automatización de pruebas funcionales.

El UFT ofrece software de automatización de pruebas funcionales y de regresión para todas las aplicaciones de software más importantes y de entorno, incluyendo avanzadas herramientas Web 2.0, principales tecnologías de desarrollo, web services…

Al grabar una prueba sobre un software con una interfaz de usuario, UFT registra las líneas de código de la prueba correspondientes con cada acción que realiza. Cada línea, por lo general, representa una acción que se lleva a cabo en un componente que aparece en la pantalla. Los componentes se llaman objetos y las acciones que se realizan en ellos se denominan operaciones.

Algunas de las principales características son las siguientes:

- Esta solución aborda las pruebas de interfaz gráfica de usuario (GUI), no GUI (API) y multicapa.

- Utiliza el lenguaje VBScript para el desarrollo y generación de los scripts de prueba y para la modificación de los mismos.

- Permite crear casos de prueba mediante la captura de flujos directamente de las pantallas de la aplicación. Además, los usuarios pueden tener acceso completo a la prueba subyacente y propiedades de los objetos.

- Permite una mayor velocidad, cobertura de la prueba y repetición de pruebas con menos recursos. Provocando también una reducción en los costes debido a la posibilidad de ejecutar pruebas 24x7 y la creación automática de documentos de prueba.

- Incluye la tecnología para la gestión de los objetos de las aplicaciones en el Gestor de Repositorio de Objetos (Object Repository Manager).

- Admite la realización de checkpoints y puntos de control para realizar comprobaciones de diferentes tipos como de características de un objeto, de bitmap, etc.

- Permite arrastrar y soltar los componentes de los procesos empresariales ya creados en su flujo de pruebas (Test Flow).

- Puede crear escenarios de automatización de pruebas sin tener que dejar la interfaz.

- Módulo de depuración y puntos de ruptura.

- Dispone de APIs para integrar soluciones terceras o de la misma suite con el objetivo de conformar una plataforma única y global (HP Application Lifecycle Management y HP LoadRunner).

- Tiene plugins adiciones que pueden ser instalados y utilizados para dar soporte y permitir la grabación y ejecución en tecnologías más específicas, como por ejemplo Flex, Citrix, Emuladores de terminal etc.

- Añade la funcionalidad de la Insight Test Automation, que se utiliza para reconocer objetos no familiares a medida que las nuevas tecnologías los incorporan.

- Posee el llamado Run Results Viewer, que se trata de un aplicativo que nos sirve para ver los resultados y/o errores de las pruebas realizadas.

HP UFT funciona sobre cualquier navegador y cualquier versión de los mismos y ofrece la compatibilidad con tecnologías más amplia del sector para todas sus necesidades de pruebas.

- **JMeter:**

El JMeter es una herramienta libre basada en el lenguaje Java, que permite realizar pruebas de Rendimiento sobre Aplicaciones Web. Es una herramienta de carga para llevar acabo simulaciones sobre cualquier recurso de Software.

Esta herramienta destaca por su versatilidad, estabilidad, y por ser de uso gratuito. También permite la ejecución de pruebas distribuidas entre distintos ordenadores para realizar pruebas de rendimiento.

Entre las características del JMETER están:

- Realizar pruebas de carga y rendimiento a diferentes tipos de servidores: Web - HTTP, HTTPS, SOAP, Database vía JDBC, LDAP, JMS, Mail - POP3(S) e IMAP(S)
- Completamente portable y 100% Java.
- Posibilidad de realizar pruebas con varios hilos (multithread).
- Interfaz (GUI) de usuario para manejo del usuario
- Análisis offline
- Estadísticas de carga
- Análisis de Data personalizable
- Manejo de script (BeanShell) para la simulación de carga

Este programa software muestra los resultados de las pruebas en una amplia variedad de informes y gráficas. Además facilita a una rápida detección de los cuellos de botella existentes debido al tiempo de respuesta excesivo.

- **HP LoadRunner:**

Se trata de una suite privada de HP dedicada a las pruebas de carga líder en el mercado de herramientas profesionales que reduce notablemente el tiempo y las competencias necesarias para generar líneas de comandos.

HP Loadrunner es una herramienta de testing de HP que ayuda a medir el rendimiento y a detectar los cuellos de botella en las aplicaciones.

Se compone de cuatro herramientas o módulos:

- Virtual User Generator (Vugen), captura los procesos de negocio de los usuarios finales y crea un script automatizado.

- El Controller, organiza, conduce, administra y supervisa la prueba de carga. Es donde diseñamos y ejecutamos los escenarios de la prueba.

- Los Load Generators (Generadores de Carga), generan la carga ejecutando los Vusers.

- Herramienta de Analysis (Analizer), que nos ayuda a ver, analizar y comparar los resultados.

Prueba una gama de aplicaciones que incluye móviles, Ajax, Flex, HTML 5, .NET, Java, GWT, Silverlight, SOAP, Citrix, ERP y elementos heredados, desde una herramienta de pruebas de software única, fácil de usar e integrada.

Permite ejecutar pruebas simples, flexibles y realistas desde múltiples geografías al escalar las pruebas de carga y descarga en la nube para simular las exigencias de las aplicaciones.

Posibilidad de integrar las pruebas de rendimiento en el entorno de desarrollo, incluidos sistemas IDE, de integración continua y construcción.

Identificar los cuellos de botella de rendimiento de las aplicaciones utilizando monitores no intrusivos de rendimiento en tiempo real que aprovechan los datos de capas de la aplicación y de nivel de código para los análisis y las causas principales.

Generación de informes y de gráficos en una interfaz amigable, dinámica y con muchas posibilidades.

Posibilidad de integrar con HP ALM para una ejecución diseñar, gestionar, ejecutar, etc., diferentes pruebas de carga. Dándonos la posibilidad de reservar intervalos de tiempo (slots) cuando realizamos la programación de una prueba desasistida.

3.2 Aplicaciones a probar.

Tras haber realizado la selección de las herramientas que vamos a utilizar para realizar las pruebas se debe de establecer el objetivo u objetivos de nuestras pruebas. Es decir, ¿qué plataforma queremos probar? ¿Qué aplicación concreta? ¿Qué prueba vamos a aplicarle? Etc.

3.2.1 Motivos y necesidades.

Tanto en el ámbito de las pruebas de rendimiento como en el de la automatización de pruebas funcionales vamos a realizar pruebas contra un entorno web, ya que todas las herramientas están preparadas para hacerlo. Se seleccionará entonces una página web que debe de tener ciertas características:

- Ser una página actual, que tenga demanda y que se encuentre en constante evolución.
- Que a pesar de tener evolutivos mantenga unas funcionalidades principales estables.
- Tener la posibilidad de realizar varias navegaciones con diferentes resultados esperados.

Con estas características, se seleccionan por lo tanto dos páginas diferentes, una más próxima y otra más comercial.

- www.uc3m.es
- www.libertyseguros.es

Sobre estas aplicaciones se diseñara un juego de casos de prueba que en el caso de las pruebas funcionales automatizadas serán las que acoten el mayor número de posibilidades y que definiremos como la "regresión" de esa página web.

También se realizarán pruebas manuales sobre las herramientas, para que podamos valorar otros aspectos de las mismas realizando un análisis funcional sobre ellas.

3.3 Plan de pruebas.

3.3.1 Pruebas cuantitativas. Definición, estrategia y alcance.

Las pruebas cuantitativas tienen como objetivo medir y calcular datos tangibles en este punto de investigación. Estos resultados serán posteriormente llevados a un análisis apropiado para así fundamentar, contrastar y realizar gráficos de los datos recogidos.

Figura 31 - Pruebas cuantitativas

Para realizar un estudio cuantitativo del sistema, arquitectura o software a probar, el proyecto se centrará en realizar dos tipos de testing automático: pruebas funcionales automatizadas y pruebas de carga.

Se harán un total de 4 pruebas funcionales automáticas sobre la página web de Liberty Seguros y 4 sobre la de la UC3M, realizando diferentes navegaciones. Además se realizará dos escenarios de pruebas de rendimiento para otra navegación sobre el portal de la UC3M, uno con menos carga de usuarios y otro con más carga de usuarios.

Gracias a estas técnicas podremos evaluar fundamentos numéricos que respalden el estudio realizado, destacando ciertos aspectos relacionados tales como:

- Tiempos de ejecución
- Iteraciones
- Usuarios virtuales
- Tiempos de respuesta
- Hits por segundo
- Consumos de memoria
- etc.

3.3.2 Pruebas cualitativas. Definición, estrategia y alcance.

Las pruebas cualitativas tienen como principal meta clasificar el objeto de estudio basándonos en características o rasgos no numéricos que son más subjetivos. Sin embargo, esos datos a pesar de no ser numéricos se pueden medir de diferentes maneras.

Esta clase de pruebas son tan importantes como las cuantitativas y nos ayudarán a abordar características del software tales como:

- Facilidad de comprensión, aprendizaje y uso
- Flexibilidad
- Exportación/Importación de datos
- Exploración de los datos
- Rapidez de las operaciones
- Ayuda y consulta usuario
- Diseño de la interfaz
- Conectividad
- etc.

Para destacar todas estas propiedades de las distintas soluciones software, por un lado realizaré pruebas manuales a modo usuario sobre las distintas herramientas para comprobar su usabilidad, potencial y amigabilidad.

Además, se confeccionarán una serie de encuestas relativas a estos aspectos y que serán entregadas a consumidores potenciales que utilizan diariamente este tipo de software en su entorno laboral. Este sondeo establecerá una numeración para poder calificar cada uno de los puntos que se exponen. Así, posteriormente podré ponerlos en común y generar un gráfico a modo de resumen.

Figura 32 - Pruebas cuantitativas

3.4 Análisis.

Este apartado especifica los requisitos, para ello se dividirán en dos tipos: requisitos de usuario de capacidad y requisitos de usuario de restricción.

3.4.1 Requisitos de usuario

En este punto se describirán tanto los requisitos de capacidad como los de restricción del usuario, recogiendo que facultades de la herramienta se van a analizar y que resultados se esperan.

Para que quede más claro y resumido, los requisitos serán recogidos en tablas que contendrán los siguientes campos:

• Nombre: Nombre identificativo del requisito. Éste será único y se utilizará a modo de resumen de dicho requisito.

• ID: identificador único del requisito de usuario. Su nomenclatura ha de seguir el formato: UR_X_ZZ, donde:

- UR indica que se trata de un requisito de usuario.
- El valor X será sustituido por '*C*' en caso de tratarse de un requisito de usuario de capacidad, o '*R*' si se tratara de un requisito de restricción.
- El valor ZZ será sustituido por el número de requisito dentro de su categoría. Comenzará en 01 e incrementará a cada nuevo requisito en una unidad.

• Necesidad: Indica como de imprescindible es que el software cumpla el requisito. Los posibles valores son '*Esencial*' y '*Deseable*'.

• Capacidad: Facultad que se estudia del software.

• Descripción: Resumen detallado de qué se aborda con el requisito.

• Resultado esperado: Cual es efecto que se pretende en la herramienta.

• Herramienta/s involucrada/s: Cuales son las herramientas que estudian dicho requisito.

Antes de mostrar los requisitos de capacidad de usuario, hay que destacar que el perfil del usuario potencial de las herramientas que se analizan son técnicos e ingenieros de pruebas, es decir, usuarios con conocimiento medio-alto de programación y alto de testing y calidad de software.

Los requisitos de usuario que se describen a continuación están sintetizados y unificados. Ya que si se describiese uno a uno las diferentes funcionalidades/capacidades de usuario que se abordan sobre las herramientas, sería desmedido debido a que algunos de los productos software analizados, sobre todo los comerciales, tienen una extensión enorme. Además los requisitos de capacidad de usuario se comprenden para todas las herramientas estudiadas, por ello no se añade el campo de herramientas involucradas.

INSTALACIÓN DEL SOFTWARE			**UR_C_01**
Necesidad	Esencial	**Capacidad**	Módulo de instalación
Descripción	El usuario debe de ser capaz de obtener e instalar de manera apropiada la herramienta.		
Resultado Esperado	Herramienta instalada en la aplicación		

Tabla 5 - Requisito de usuario UR_C_01

ACCEDER AL SOFTWARE			**UR_C_02**
Necesidad	Esencial	**Capacidad**	Módulo de arranque
Descripción	El usuario debe de ser capaz de acceder correctamente al software.		
Resultado Esperado	Visualizar el panel principal de la herramienta.		

Tabla 6 - Requisito de usuario UR_C_02

REALIZAR GRABACIÓN WEB			**UR_C_03**
Necesidad	Esencial	**Capacidad**	Módulo de grabación
Descripción	El usuario debe de ser capaz de grabar un caso de prueba sobre una plataforma web.		
Resultado Esperado	Se genera un script en el lenguaje de la herramienta, que se compone de los pasos deseados.		

Tabla 7 – Requisito de usuario UR_C_03

REALIZAR GRABACIÓN NO WEB			UR_C_04
Necesidad	Deseable	**Capacidad**	Módulo de grabación
Descripción	El usuario debe de ser capaz de grabar un caso de prueba sobre una plataforma no web.		
Resultado Esperado	Se genera un script en el lenguaje de la herramienta, que se compone de los pasos deseados.		

Tabla 8 – Requisito de usuario UR_C_04

MODIFICAR SCRIPTS			UR_C_05
Necesidad	Esencial	**Capacidad**	Módulo de depuración
Descripción	El usuario debe de ser capaz de depurar y modificar un script ya generado.		
Resultado Esperado	Se obtiene un caso de prueba modificado y que cumple las expectativas deseadas.		

Tabla 9 – Requisito de usuario UR_C_05

REALIZAR EJECUCIÓN			UR_C_06
Necesidad	Esencial	**Capacidad**	Módulo de ejecución
Descripción	El usuario debe de ser capaz de ejecutar el caso de prueba escogido.		
Resultado Esperado	Se lanza un script a través la herramienta y se reproduce la navegación o funcionalidad que recoge.		

Tabla 10 - Requisito de usuario UR_C_06

ANÁLISIS DE RESULTADOS			UR_C_07
Necesidad	Esencial	**Capacidad**	Módulo de ejecución
Descripción	El usuario debe de ser capaz de ver y analizar los resultados de la prueba.		
Resultado Esperado	Se generan resultados que se muestran en diferentes formatos y que son susceptibles de análisis.		

Tabla 11- Requisito de usuario UR_C_07

AYUDA			UR_C_08
Necesidad	Esencial	**Capacidad**	Módulo de cooperación
Descripción	El usuario debe de ser capaz de utilizar un apartado de ayuda y de asistencia técnica como apoyo.		
Resultado Esperado	Se muestra una sección de Ayuda al usuario en la que se puede encontrar material aprovechable.		

Tabla 12 - Requisito de usuario UR_C_08

ASISTENCIA TÉCNICA			UR_C_09
Necesidad	Deseable	**Capacidad**	Módulo de cooperación
Descripción	El usuario dispone de asistencia técnica como cliente de la herramienta.		
Resultado Esperado	Se puede contactar con el servicio técnico mediante medios aportados.		

Tabla 13 - Requisito de usuario UR_C_09

Los requisitos de restricción, indican qué limitaciones tienen las pruebas. A continuación se detallan las restricciones generales que se desprenden a partir de las herramientas estudiadas, aunque no se cumplan en todas ellas:

IMPOSIBILIDAD DE MULTISESIÓN			UR_R_01
Necesidad	Deseable	**Capacidad**	Módulo de acceso
Descripción	El usuario intenta ejecutar a la vez 2 o más veces una herramienta.		
Resultado Esperado	El usuario no puede abrir más de una sesión de la herramienta.		
Herramienta/s involucrada/s	HP UFT, Selenium IDE		

Tabla 14 - Requisito de usuario UR_R_01

IMPOSIBILIDAD DE UTILIZAR SIN NOCIONES DE PROGRAMACIÓN			UR_R_02
Necesidad	Esencial	**Capacidad**	Módulo de grabación y depuración
Descripción	El usuario no es capaz de realizar casos de prueba consistentes, si no conoce el idioma de desarrollo en que se implementa.		
Resultado Esperado	El usuario no tiene capacidad para realizar cambios y depuración del script.		
Herramienta/s involucrada/s	HP UFT, Selenium IDE, HP Loadrunner, Apache Jmeter		

Tabla 15 - Requisito de usuario UR_R_02

IMPOSIBILIDAD DE PROGRAMAR EN VARIOS LENGUAJES			UR_R_03
Necesidad	Deseable	**Capacidad**	Módulo de grabación y depuración
Descripción	El usuario intenta desarrollar un script en 2 lenguajes distintos sobre la misma herramienta.		
Resultado Esperado	El usuario no puede implementar sobre una herramienta, casos de prueba en diferentes idiomas de programación.		
Herramienta/s involucrada/s	HP UFT, HP Loadrunner		

Tabla 16 - Requisito de usuario UR_R_03

IMPOSIBILIDAD DE EJECUTAR EN REMOTO			UR_R_04
Necesidad	Deseable	**Capacidad**	Módulo de ejecución
Descripción	El usuario intenta ejecutar remotamente la prueba en otra máquina.		
Resultado Esperado	El usuario sólo puede ejecutar en localhost.		
Herramienta/s involucrada/s	HP UFT, Selenium IDE		

Tabla 17 - Requisito de usuario UR_R_04

POSIBILIDAD DE INTERFERIR EN LAS PRUEBAS			UR_R_05
Necesidad	Deseable	**Capacidad**	Módulo de ejecución
Descripción	El usuario al realizar movimientos de cursor, tecleos, navegaciones etc., es capaz de interferir sobre la ejecución de la prueba.		
Resultado Esperado	El usuario modifica el funcionamiento real del caso de prueba.		
Herramienta/s involucrada/s	HP UFT, Selenium IDE		

Tabla 18 - Requisito de usuario UR_R_05

IMPOSIBILIDAD DE EXPORTAR RESULTADOS EN UN FORMATO PROPIO			UR_R_06
Necesidad	Deseable	**Capacidad**	Módulo de análisis
Descripción	El usuario no es capaz de visualizar los resultados en un formato propio.		
Resultado Esperado	No se puede trasladar los resultados obtenidos en la prueba a una forma personalizada.		
Herramienta/s involucrada/s	HP UFT, Selenium IDE, HP Loadrunner, Apache Jmeter		

Tabla 19 - Requisito de usuario UR_R_06

3.4.2 Requisitos de software

En este punto se describirá los requisitos técnicos del software, y se dividirán en dos categorías: funcionales y no funcionales. Estos se abordan de una manera resumida ya que los requisitos técnicos de las diferentes herramientas son exagerados.

Para que quede más claro y resumido, los requisitos serán recogidos en tablas que contendrán los siguientes campos:

• Nombre: Nombre identificativo del requisito. Éste será único y se utilizará a modo de resumen de dicho requisito.

• ID: identificador único del requisito técnico. Su nomenclatura ha de seguir el formato: SR_XX_ZZ, donde:

- SR indica que se trata de un requisito de software/técnico.
- El valor XX será sustituido por '*RF*' en caso de tratarse de un requisito funcional, o '*NF*' si se tratara de un requisito no funcional.
- El valor ZZ será sustituido por el número de requisito dentro de su categoría. Comenzará en 01 e incrementará a cada nuevo requisito en una unidad.

• Necesidad: Indica como de imprescindible es que el software cumpla el requisito. Lo posibles valores son '*Esencial*' y '*Deseable*'.

• Capacidad: Facultad que se estudia del software.

• Descripción: Resumen detallado de que se aborda con el requisito.

• Resultado esperado: Cual es efecto que se pretende en la herramienta.

• Herramienta/s involucrada/s: Cuales son las herramientas que estudian dicho requisito.

• Dependencias UR: Indica qué requisito de usuario se evalúa o implementa con dicho requisito software. Cada requisito de usuario tiene que estar cubierto por uno o varios requisitos software.

REPOSITORIO DE OBJETOS			**SR_RF_01**
Necesidad	Deseable	**Capacidad**	Módulo de grabación y depuración
Descripción	Se tiene acceso a un repositorio de objetos donde se puede ver y editar todos los objetos grabados.		
Resultado Esperado	Aparece una interfaz de usuario de repositorio de objetos		
Herramienta/s involucrada/s	HP UFT		
Dependencias UR	UR_C_02, UR_C_03, UR_C_04, UR_C_05		

Tabla 20 - Requisito de software SR_RF_01

MONITORIZACIÓN EN TIEMPO REAL			SR_NF_02
Necesidad	Esencial	**Capacidad**	Módulo de ejecución y análisis
Descripción	Se tiene pueden ver la monitorización de recursos de las máquinas involucradas. Viendo CPU, % memoria consumida, tiempos etc.		
Resultado Esperado	Se ven los recursos consumidos por la infraestructura que toma parte en las pruebas.		
Herramienta/s involucrada/s	HP Loadrunner, Apache Jmeter		
Dependencias UR	UR_C_02, UR_C_06, UR_C_07, UR_R_04		

Tabla 21 - Requisito de software SR_NF_02

GRABACIÓN EN MODO CLICK-AND-SCRIPT			SR_RF_03
Necesidad	Esencial	**Capacidad**	Módulo de grabación
Descripción	Se permite realizar scripts básicos utilizando el método de click and script.		
Resultado Esperado	Graba scripts básicos automáticos realizando una navegación/funcionalidad de manera manual.		
Herramienta/s involucrada/s	HP UFT, Selenium IDE, HP Loadrunner, Apache Jmeter		
Dependencias UR	UR_C_02, UR_C_03, UR_C_04,		

Tabla 22 - Requisito de software SR_RF_03

BOTÓN AYUDA			SR_RF_04
Necesidad	Esencial	**Capacidad**	Módulo de cooperación
Descripción	Al pulsar el botón ayuda vemos una serie de documentos y registros de apoyo.		
Resultado Esperado	El usuario puede buscar información y guías de ayuda asociadas a la herramienta.		
Herramienta/s involucrada/s	HP UFT, Selenium IDE, HP Loadrunner, Apache JMeter		
Dependencias UR	UR_C_02, UR_C_08, UR_C_09		

Tabla 23 - Requisito de software SR_RF_04

UTILIZACIÓN MULTI-PLATAFORMA			SR_NF_05
Necesidad	Esencial	**Capacidad**	Módulo de instalación
Descripción	Instalar y utilizar la herramienta en diferentes sistemas operativos, Windows, UNIX etc.		
Resultado Esperado	Se permite su implementación en tales plataformas.		
Herramienta/s involucrada/s	Selenium IDE, Apache Jmeter		
Dependencias UR	UR_C_01, UR_C_02		

Tabla 24 - Requisito de software SR_NF_05

UTILIZACIÓN MULTI-IDIOMA			SR_NF_06
Necesidad	Esencial	**Capacidad**	Módulo de grabación
Descripción	Se permite la utilización de la herramienta en diferentes idiomas de desarrollo.		
Resultado Esperado	Se puede desarrollar e implementar scripts a varios idiomas de programación.		
Herramienta/s involucrada/s	Selenium IDE, Apache Jmeter		
Dependencias UR	UR_C_02, UR_C_03, UR_C_04, UR_C_05, UR_R_02, UR_R_03		

Tabla 25 - Requisito de software SR_NF_06

ENCRIPTACIÓN DE DATOS			SR_NF_07
Necesidad	Esencial	**Capacidad**	Módulo de grabación y depuración
Descripción	Se permite codificar datos vulnerables, como contraseñas.		
Resultado Esperado	Esta encriptación se realiza aportando un valor de seguridad apropiado al script.		
Herramienta/s involucrada/s	HP UFT, HP Loadrunner		
Dependencias UR	UR_C_02, UR_C_03, UR_C_04, UR_C_05		

Tabla 26 - Requisito de software SR_NF_07

VISTA EXECUTION-FLOW			SR_RF_08
Necesidad	Esencial	**Capacidad**	Módulo de grabación y ejecución
Descripción	Se permite ver el flujo de ejecución que seguirá el script de manera gráfica.		
Resultado Esperado	Se ven el Execution Flow.		
Herramienta/s involucrada/s	HP UFT		
Dependencias UR	UR_C_02, UR_C_05, UR_C_06		

Tabla 27 - Requisito de software SR_RF_08

ENTRADA/SALIDA DE DATOS			SR_NF_09
Necesidad	Esencial	**Capacidad**	Módulo de grabación, depuración y ejecución
Descripción	Entrada/Salida mediante ficheros, tablas, parámetros etc.		
Resultado Esperado	Se pueden parametrizar los datos de entrada o salida del script a través de diferentes medios.		
Herramienta/s involucrada/s	HP UFT, HP LoadRunner		
Dependencias UR	UR_C_02, UR_C_03, UR_C_04, UR_C_05		

Tabla 28 - Requisito de software SR_NF_09

UTILIZACIÓN PARA VARIOS PROTOCOLOS			SR_NF_10
Necesidad	Esencial	**Capacidad**	Módulo de grabación, depuración y ejecución
Descripción	Permite la grabación en distintos protocolos.		
Resultado Esperado	Se trabaja correctamente con una variedad de protocolos (Web, Rdp, Citrix,..) y formatos (imágenes, mapas, etc.)		
Herramienta/s involucrada/s	HP UFT, HP LoadRunner, Apache Jmeter		
Dependencias UR	UR_C_02, UR_C_03, UR_C_04, UR_C_05, UR_C_06, UR_R_02		

Tabla 29 - Requisito de software SR_NF_10

INTERCAMBIAR RECURSOS			SR_RF_11
Necesidad	Esencial	**Capacidad**	Módulo de grabación y depuración
Descripción	Se permite abrir dos scripts y copiar datos de una a otro, manteniendo la funcionalidad intacta.		
Resultado Esperado	Se puede trabajar y reutilizar código y recursos.		
Herramienta/s involucrada/s	HP Loadrunner, Apache Jmeter		
Dependencias UR	UR_C_02, UR_C_03, UR_C_04, UR_C_05, UR_R_01, UR_R_02		

Tabla 30 - Requisito de software SR_RF_11

LOG DE RESULTADOS Y ERRORES			SR_RF_12
Necesidad	Esencial	**Capacidad**	Módulo de análisis
Descripción	Se devuelve un log o informe con los resultados de las pruebas realizadas.		
Resultado Esperado	Este log es predecible y analizable por el usuario.		
Herramienta/s involucrada/s	HP UFT, Selenium IDE, HP Loadrunner, Apache Jmeter		
Dependencias UR	UR_C_02, UR_C_07, UR_R_05, UR_R_06		

Tabla 31 - Requisito de software SR_RF_12

RESULT VIEWER Y GRÁFICOS DE RESULTADOS			SR_RF_13
Necesidad	Esencial	**Capacidad**	Módulo de análisis
Descripción	Se devuelve un resumen completo y gráficos de los resultados		
Resultado Esperado	Aporta gráficos y tablas de resultados con información valorable.		
Herramienta/s involucrada/s	HP UFT, HP Loadrunner, Apache Jmeter		
Dependencias UR	UR_C_07, UR_R_06		

Tabla 32 - Requisito de software SR_RF_13

SNAPSHOTS			SR_RF_14
Necesidad	Esencial	**Capacidad**	Módulo de ejecución y análisis
Descripción	Se realizan capturas de pantalla e incluso vídeos durante la ejecución de la prueba.		
Resultado Esperado	Se puede ver los snapshots y/o Screen recorders que se generan y utilizarse como evidencias.		
Herramienta/s involucrada/s	HP UFT, HP Loadrunner		
Dependencias UR	UR_C_06,UR_C_07, UR_R_06		

Tabla 33 - Requisito de software SR_RF_14

CAPACIDAD DE CARGA DE LA PRUEBA			SR_NF_15
Necesidad	Esencial	**Capacidad**	Módulo de ejecución
Descripción	Capacidad para realizar una carga de Vusers o grupo de hilos.		
Resultado Esperado	Se puede realizar la carga hasta el límite deseado por el usuario.		
Herramienta/s involucrada/s	HP Loadrunner, Apache Jmeter		
Dependencias UR	UR_C_06, UR_R_04		

Tabla 34 - Requisito de software SR_NF_15

INTERFAZ Y CONFIGURACIÓN DEL ESCENARIO			SR_NF_16
Necesidad	Esencial	**Capacidad**	Módulo de ejecución
Descripción	Posibilidades de configuración y ejecución del escenario.		
Resultado Esperado	Se puede introducir una rampa de entrada, una de salida, tiempo de estabilidad, pacings, think times, etc.		
Herramienta/s involucrada/s	HP Loadrunner, Apache Jmeter		
Dependencias UR	UR_C_06, UR_R_04		

Tabla 35 - Requisito de software SR_NF_16

3.4.3 Matriz de trazabilidad

Tras recoger los requisitos de usuario y los requisitos del software, se debe de confirmar que todo requisito de usuario esté contemplado por al menos un requisito software.

Para hacer esta comprobación, se utilizan matrices de trazabilidad. Una matriz de trazabilidad tiene en su eje vertical el conjunto de identificadores de requisitos de usuario, tanto de capacidad como de restricción, y en su eje horizontal, los identificadores de requisitos software.

	SR_RF_01	SR_NF_02	SR_RF_03	SR_RF_04	SR_NF_05	SR_NF_06	SR_NF_07	SR_RF_08	SR_NF_09	SR_NF_10	SR_RF_11	SR_RF_12	SR_RF_13	SR_RF_14	SR_NF_15	SR_NF_16
UR_C_01					X											
UR_C_02	X	X	X	X	X	X	X	X	X	X	X	X				
UR_C_03	X		X			X	X		X	X	X					
UR_C_04	X		X			X	X		X	X	X					
UR_C_05	X					X	X	X	X	X	X					
UR_C_06		X						X		X				X	X	X
UR_C_07		X										X	X	X		
UR_C_08				X												
UR_C_09				X												
UR_R_01											X					
UR_R_02						X				X	X					
UR_R_03						X										
UR_R_04		X													X	X
UR_R_05												X				
UR_R_06												X	X	X		

Tabla 36 - Matriz trazabilidad requisitos usuario y técnicos

Capítulo 4

Diseño, implementación y ejecución de las pruebas.

En este capítulo, se realiza en primer lugar el diseño de los casos de prueba, cuyo contenido quedará descrito mediante tablas con los diferentes campos descriptivos.

A continuación, se realiza la implementación en los diferentes útiles software de los que disponemos. Esto dará lugar a scripts y/o escenarios de ejecución con las funcionalidades deseadas.

Por último, se ejecutan las pruebas automáticas, gracias a las cuales se alcanzarán los objetivos deseados y se reunirán los resultados obtenidos para el estudio venidero.

4.1 Pruebas manuales.

Realizaré una serie de pruebas exploratorias para resaltar y llevar a estudio el funcionamiento, facetas, necesidades y componentes más característicos y útiles para un usuario de este tipo de herramientas.

- **Diseño:**

Los casos de prueba definidos para realizar las pruebas manuales tienen como finalidad ver los detalles de interfaz y funcionales de las herramientas automáticas. Estas pruebas se realizan con el fin de, según haya sido la experiencia sobre cada plataforma, darle un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Nada satisfactorio	Poco satisfactorio	Relativamente satisfactorio	Bastante satisfactorio	Muy satisfactorio
1	2	3	4	5

Tabla 37 - Grados satisfacción

Por tanto, los casos de prueba se mostrarán en tablas y se compondrán de los siguientes campos:

- ID: identificador de la prueba
- Nombre: título del caso de prueba
- Tipo de prueba: Funcional, Regresión, Carga etc.
- Descripción: Qué objetivo pretende el caso de prueba
- Pasos: Acciones a seguir
- Criterio de fallo: Cuándo se declarará fallida la prueba
- Aplicación/es: Aplicación/es a probar

Todas las pruebas manuales se repetirán 4 veces, ya que se hará una por cada herramienta que utilizamos.

Tras su implementación y ejecución, en la fase de análisis de los resultados se especificará el valor correspondiente al campo Resultado, ya que éste nos ofrece la respuesta derivada de la prueba.

PF001	Instalación herramienta
Tipo Prueba	Prueba Funcional
Descripción	Comprobar requisitos mínimos necesarios y nivel de complejidad para instalar el software de automatización en el equipo.
Pasos	1. Comprobar requisitos del sistema 2. Descargar el software 3. Instalar en el equipo 4. Probar que se abre correctamente
Criterio de fallo	La instalación no se realiza correctamente
Aplicación/es	HP UFT, Selenium, HP LoadRunner, JMeter

Tabla 38 - Prueba manual PF001

PF002	Interacción con la GUI
Tipo Prueba	Prueba Funcional
Descripción	Comprobar lo amigable, potente e intuitiva que es la interfaz del usuario en el programa automático.
Pasos	1. Interactuar con la interfaz y realizar diferentes movimientos por la misma fijándonos en los menús, barras, opciones, estructura, etc., del mismo.
Criterio de fallo	No tiene Interfaz gráfica de usuario
Aplicación/es	HP UFT, Selenium, HP LoadRunner, JMeter

Tabla 39 - Prueba manual PF002

PF003	Ayuda y Consulta de usuario
Tipo Prueba	Prueba Funcional
Descripción	Comprobar la disponibilidad de guías y/o ayuda integrada y correspondiente al software para que consulte el usuario.
Pasos	1.Buscar ayuda y/o guías 2.Ver su extensión y posibilidades
Criterio de fallo	No existe ayuda de usuario
Aplicación/es	HP UFT, Selenium, HP LoadRunner, JMeter

Tabla 40 - Prueba manual PF003

PF004	Integración con otras herramientas
Tipo Prueba	Prueba Funcional
Descripción	Realizar pruebas de las posibilidades de conectar e integrar este software con otras herramientas que la complementen.
Pasos	1. Realizar conexión con otra herramienta 2. Comprobar funcionalidad conjunta
Criterio de fallo	No existe posibilidad de integrarla con otra herramienta.
Aplicación/es	HP UFT, Selenium, HP LoadRunner, JMeter

Tabla 41 - Prueba manual PF004

PF005	Grabación, reutilización y depuración
Tipo Prueba	Prueba Funcional
Descripción	Realizar pequeñas pruebas de grabación, comprobar las posibilidades de reutilizar scripts y parametrizar valores, y ver tanto la facilidad de completarlos como de depurarlos.
Pasos	1. Grabar un script nuevo 2. Realizar cambios en el script
Criterio de fallo	No tiene opción de mantener y depurar el script creado
Aplicación/es	HP UFT, Selenium, HP LoadRunner, JMeter

Tabla 42 - Prueba manual PF005

PF006	Ejecución
Tipo Prueba	Prueba Funcional
Descripción	Ejecutar pequeñas pruebas y estudiar las maneras de hacerlo.
Pasos	1. Ejecutar un script 2. Ver los modos y características de la ejecución
Criterio de fallo	No tiene opción de configurar y ejecutar scripts a través de la interfaz
Aplicación/es	HP UFT, Selenium, HP LoadRunner, JMeter

Tabla 43 - Prueba manual PF006

PF007	Análisis y Reporting
Tipo Prueba	Prueba Funcional
Descripción	Ver las posibilidades que existen a la hora de ver resultados y analizarlos.
Pasos	1. Ver resultados. 2. Posibilidad de reportar resultados.
Criterio de fallo	No tiene opción de ver informe con los resultados alcanzados
Aplicación/es	HP UFT, Selenium, HP LoadRunner, JMeter

Tabla 44 - Prueba manual PF007

- **Ejecución:**

  - **PF001: Instalación herramienta**

En primer lugar se descargan las últimas versiones desde las suites oficiales de HP, Selenium y Apache JMeter. Para ello, se nos pide un registro en las herramientas de pago y adjunta una licencia al producto que permite su utilización gratuita por el periodo de 30 días.

Antes de realizar la instalación de las distintas herramientas debemos de fijarnos y cumplir los requisitos mínimos del sistema para cada uno de ellos.

	Mínimos requeridos JMeter 2.13
Máquina Virtual de JAVA	Versión 6 o superior
Sistema Operativo	· Windows 7 Oracle JDK 7 64-bit · Mac OSX 10.9.X JDK 6 64-bit · Mac OSX 10.9.X JDK 7u71 / 8u20 & 8u25 64-bit

Tabla 45 - Requisitos mínimos JMeter 2.13

	Mínimos requeridos HP LoadRunner 12.50
Procesador	Dual Core 2.2 Ghz o superior
Sistema Operativo	· Windows Server 2008 R2 SP1 64-bit · Windows Server 2012 R2 64-bit · Windows 7 SP1 32 o 64-bit · Windows 8.1 64-bit
Memoria (RAM)	Recomendado 8 GB
Resolución de pantalla	Mínimo: 1366X768
Espacio disponible en el disco duro	Mínimo: 50 GB
Navegador	Microsoft Internet Explorer 9, 10 o 11 (Es recomendable con la configuración por defecto de IE)

Tabla 46 - Requisitos mínimos HP LoadRunner 12.50

	Mínimos requeridos Selenium IDE 2.9
Mozilla Firefox	Versión 2 o superior
Sistema Operativo	Windows, OS X, Linux, Solaris con versión 2 o superior de Firefox

Tabla 47 - Requisitos mínimos Selenium IDE 2.9

	Mínimos requeridos HP UFT 12.51
Procesador	1.6 Ghz o superior
Sistema Operativo	Windows 7 Service Pack 1 (32 bit o 64 bit)
Memoria	Mínimo 2 GB si no hay más de 3 add-ins cargados simultáneamente Nota: · 512 MB de RAM adicional cuando se usa una máquina virtual. · Se requiere memoria adicional cuando se cargan más add-ins y cuando se quieren guardar los vídeos generados para los informes de resultados durante las ejecuciones.
Disco duro	5400 RPM
Ajustes de Color	Alta densidad de color (16 bit)
Tarjeta Gráfica	Tarjeta gráfica de 64 MB memoria de vídeo
Espacio disponible en el disco duro	2 GB de disco duro libre para aplicaciones de archivos y carpetas Adicionalmente, se debe tener 1GB de disco libre en el disco de sistema para su utilización.
Navegador	Microsoft Internet Explorer 8.0 o versión posterior

Tabla 48 - Requisitos mínimos HP UFT 12.51

Las necesidades para la puesta en marcha de las herramientas comerciales suponen una mayor cantidad y calidad de los requisitos del sistema (memoria RAM, espacio libre en disco duro etc.), es decir requieren más recursos y que el software necesario esté más actualizado. Siendo HP LoadRunner el que más cantidad de recursos necesita.

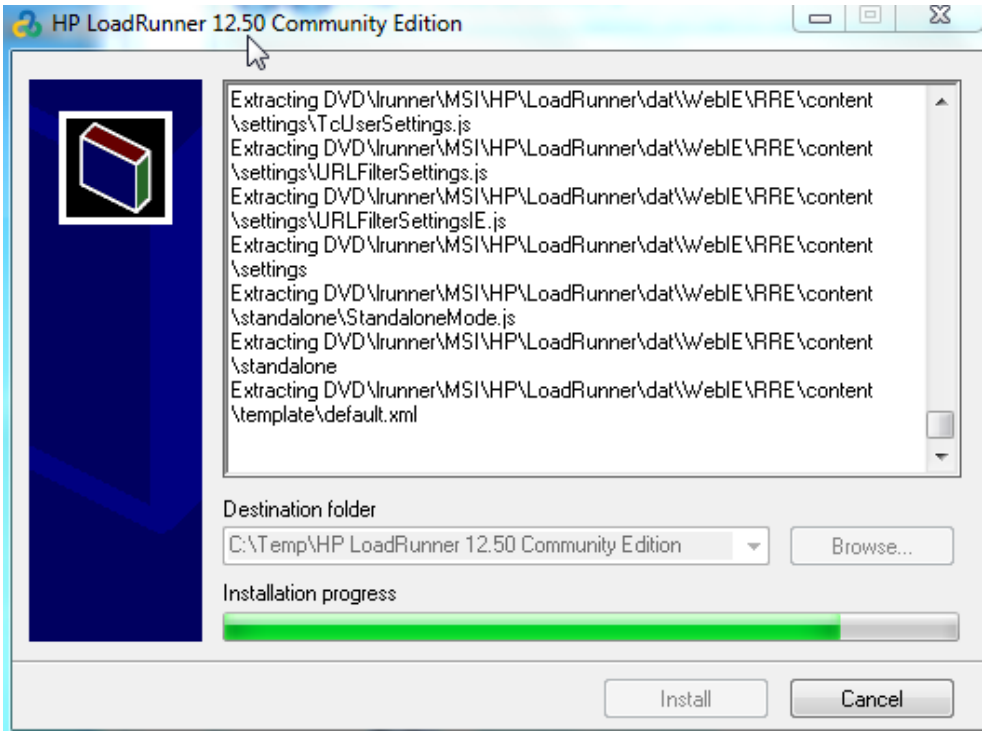


Figura 33 - Instalación HPLoadRunner

En consecuencia, el proceso de instalación es más pesado y largo para las herramientas de la suite de HP ya que instalan más módulos y son más grandes que los de las gratuitas.

- **PF002: Interacción con la GUI**

Se realizan diversas navegaciones por las diferentes herramientas y se observa que la interfaz de usuario de las soluciones de HP tiene un denominador común; y es que tienen una apariencia más agradable e intuitiva para que el usuario interactúe de una manera más eficiente y sencilla.

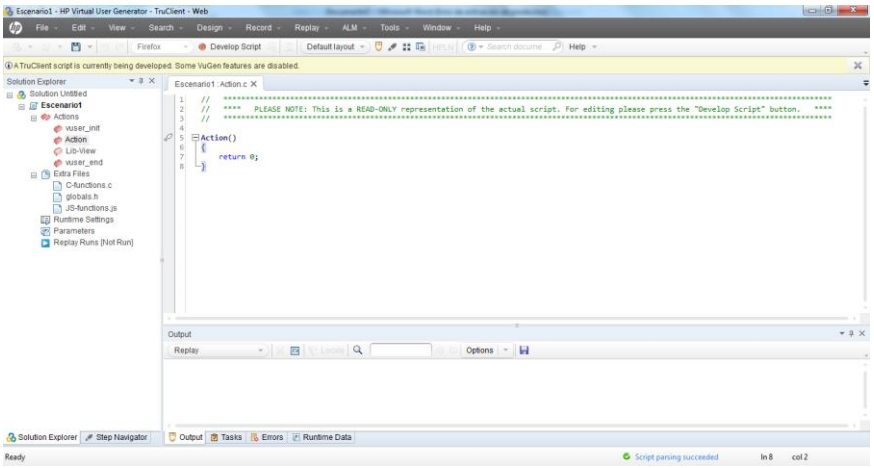


Figura 34 – Interfaz VuGen HP LoadRunner

Sin embargo, las gratuitas a pesar de que su interfaz es menos vistosa es igual de eficiente, puesto que las limitaciones de ésta están ligadas a las limitaciones que pueda tener la herramienta.

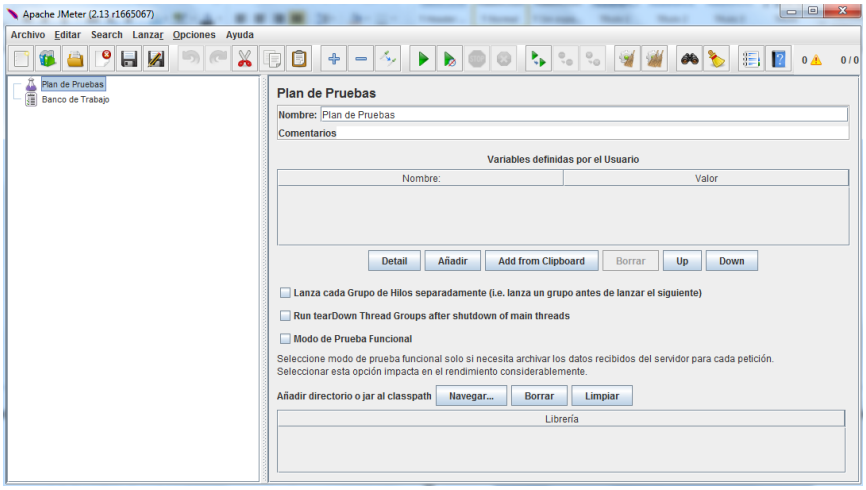


Figura 35 - Interfaz Apache JMeter

Por tanto, la calidad de las interfaces son claramente superiores y mucho más amigables en los productos comerciales que en las herramientas libres. Esto se debe a que en las privativas se le da más importancia a la imagen y a que el dinero que se invierte en ello es mucho mayor que en las gratuitas, puesto que en éstas lo primordial es su funcionalidad ya que no dispone de los medios económicos que tienen los privados.

- **PF003: Ayuda y Consulta de usuario**

Las cuatro herramientas tienen una sección dedicada a la ayuda del usuario. Sin embargo la ayuda de Selenium IDE es de tipo comunidad y sólo online, mientras que los otros productos llevan la ayuda incorporada.

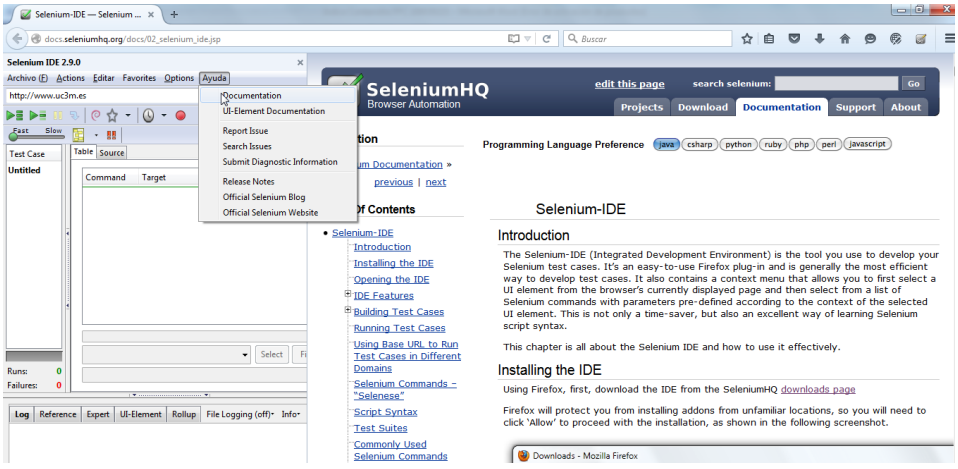


Figura 36 - Ayuda Selenium IDE

Las herramientas de pago son las que tienen más posibilidades dentro de la ayuda y apoyo al usuario. Ya que tienen, por ejemplo un buscador de términos, vídeos de ayuda, foros de consultas etc.

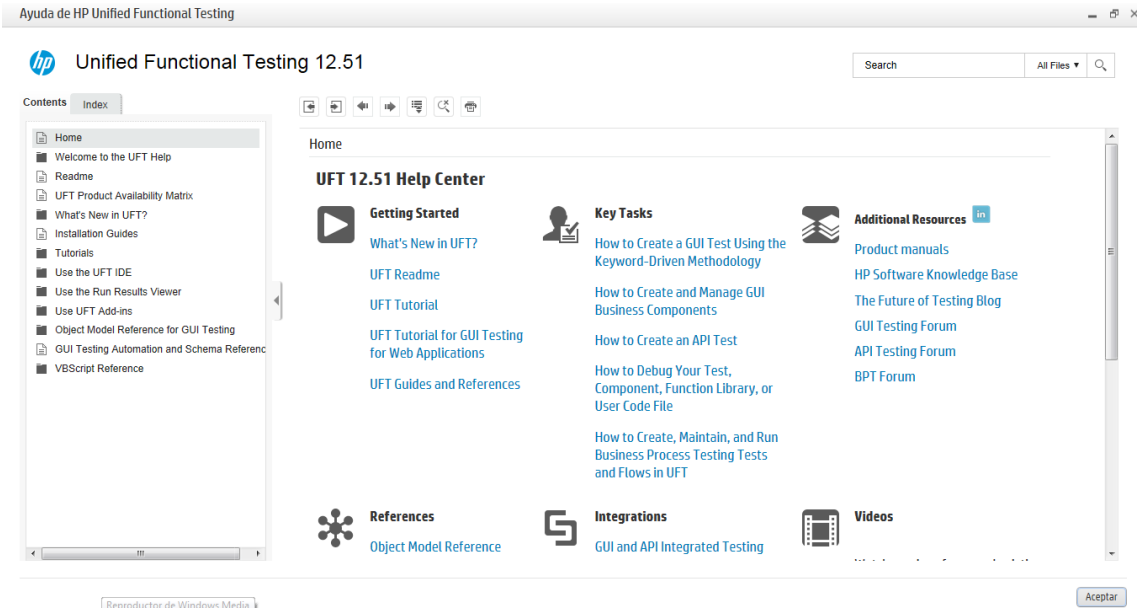


Figura 37 - Ayuda HP UFT

También se observa que tanto HP UFT como HP Loadrunner tienen servicio de soporte, mientras que las gratuitas no lo tienen.

- **PF004: Integración con otras herramientas**

Los productos de HP de pruebas se pueden integrar entre sí unificándose gracias a Quality Center (ALM), que es la herramienta de Gestión de Calidad que permite la administración, ejecución, análisis, reporting, etc., de una manera centralizada.

Jmeter dispone de un plugin para integrarse con Eclipse. Otra opción es ejecutarse por línea de comandos, posibilitando el hacerlo desde otras aplicaciones, pero no llegando ser una integración al uso. También se puede es ejecutar a través de una tarea ANT planes de prueba en esta solución y reportar los resultados en un fichero.

Selenium se integra con Jenkins para poder obtener informes de resultados, ya que por sí sólo no dispone de ningún módulo de análisis y reporting. Jenkins también permitirá darle mayor potencial a la hora de grabar y ejecutar los casos de prueba en diferentes navegadores y S.O. Maven, Testlink, Hudson etc., son otras soluciones software que se pueden integrar con Selenium.

Por tanto, las herramientas de HP aunque están más experimentadas y son más estables y robustas, siempre se integrarán con herramientas de la misma propiedad. Sin embargo, los productos libres están avanzando continuamente y gracias a los desarrolladores se crean continuamente nuevos módulos para integrarse con gran variedad de software existente en el mercado.

- **PF005: Grabación, reutilización y depuración**

Tanto en las herramientas de HP como en Selenium se permite el "Click and Script" es decir grabar una navegación y generar automáticamente líneas de código. Sin embargo, éstas luego tienen que ser tratadas y modificadas a nuestro gusto. Esto es relativamente sencillo en las herramientas de HP y algo más laborioso en Selenium IDE.

HP nos aporta diferentes añadidos que ayudan al usuario a la hora de grabar y mantener un script. Por ejemplo, en HP UFT con la existencia de un repositorio de objetos donde se guardan todos los elementos con los que hemos interaccionado a lo largo de la grabación, siendo muy útil.

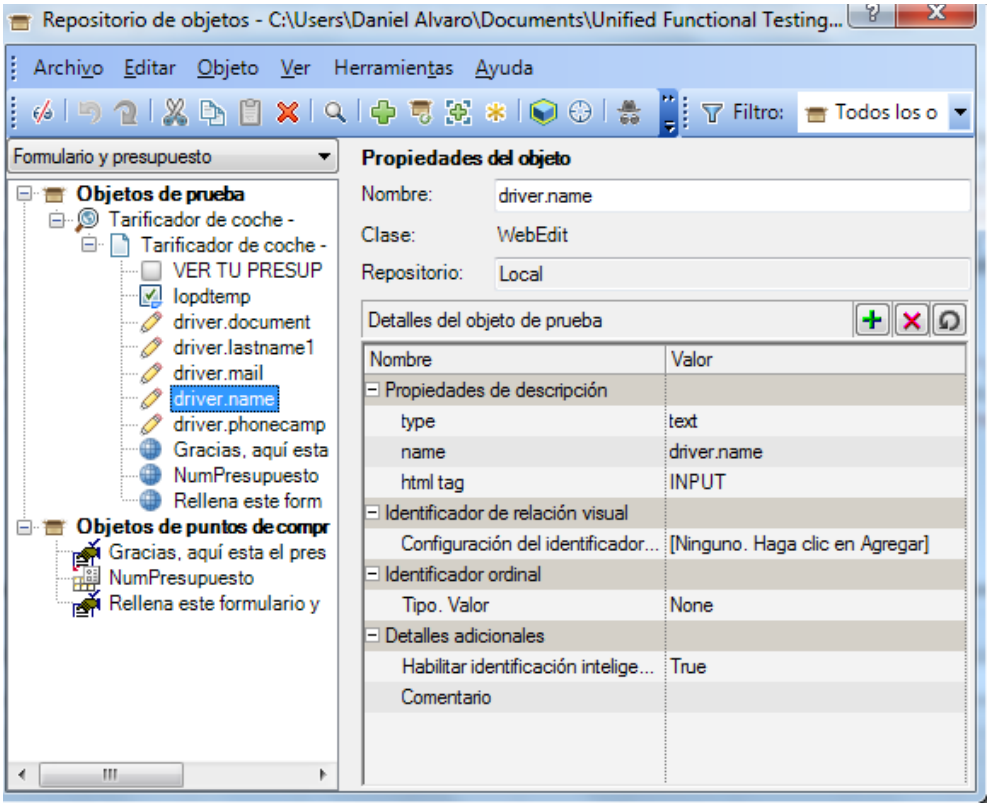


Figura 38 - Repositorio HP UFT

También cabe destacar que UFT dispone de mucho add-ins para automatizar diferentes arquitecturas y plataformas (Siebel, Java, etc.).

En el VuGen de HP LoadRunner aparece el protocolo TruCLient y éste se puede utilizar, a partir de la versión 12.50, sobre cualquier navegador. Es un protocolo muy interactivo, agradable y más sencillo de utilizar y mantener que el típico desarrollo con código http.

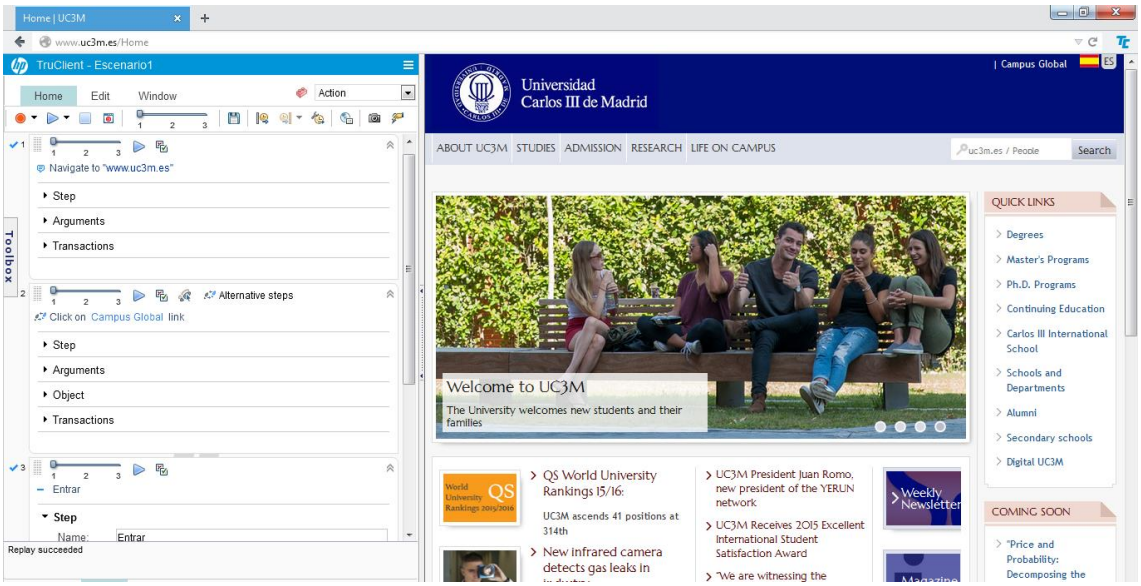


Figura 39 - Protocolo TruClient

Tanto Selenium como las herramientas de HP nos permiten tanto interactuar como modificar el script en tiempo de grabación el script. Sin embargo, Selenium no nos permite encriptar claves o passwords mientras que las herramientas de pago si lo permiten.

JMeter por defecto tan sólo nos permite realizar pruebas de carga pero haciendo llamadas a la página web, sin embargo, es capaz de realizar simulaciones de navegaciones y capturar el tráfico generado.

- **PF006: Ejecución**

La ejecución de casos de prueba en UFT es sencilla y hay que tener en cuenta que la ejecución al realizarse en local, no debemos de realizar operaciones ni movimientos sobre dicha máquina ya que podríamos interferir en la prueba llevándolo a un resultado fallido. Se nos permite la ejecución de varias iteraciones, cada una con distintos datos de entrada etc.

Selenium está diseñado para realizarse en una sola ejecución y que corriesen todos los casos de prueba deseados de manera secuencial.

La ejecución en LoadRunner de los escenarios de prueba se realiza mediante el Controller. Una vez abierto el escenario implementado y configurado previamente, se ejecuta realizando un simple click que desata la ejecución de los casos de prueba. Se verá en tiempo de ejecución la carga de usuarios virtuales que entran o salen, las transacciones correctas y fallidas, etc. También observamos gráficas actualizándose en tiempo de ejecución con recursos de las infraestructuras, transacciones por segundo, rampa de usuarios etc. La visualización de estas gráficas se puede configurar.

En JMeter la ejecución de los escenarios de prueba es más sencilla. Se pulsa el botón de reproducción ejecutando el grupo de hilos diseñados y se pueden utilizar bucles, condicionantes y configurar para que se ejecute hasta un número ilimitado de veces.

- **PF007: Análisis y Reporting**

Los útiles comerciales nos posibilitan un reporte de los resultados obtenidos en la prueba que nos aportan información y una visión específica de cómo ha ido la prueba realizada.

En HP UFT tenemos el "Result Viewer" donde nos encontramos con información útil como un resumen global, las iteraciones llevadas a cabo, los componentes sobre los que se ha actuado, chequeos realizados etc. Estos resultados aparecen en forma de árbol y pueden ir acompañados tanto de imágenes como del vídeo de la ejecución según hayamos configurado previamente la herramienta.

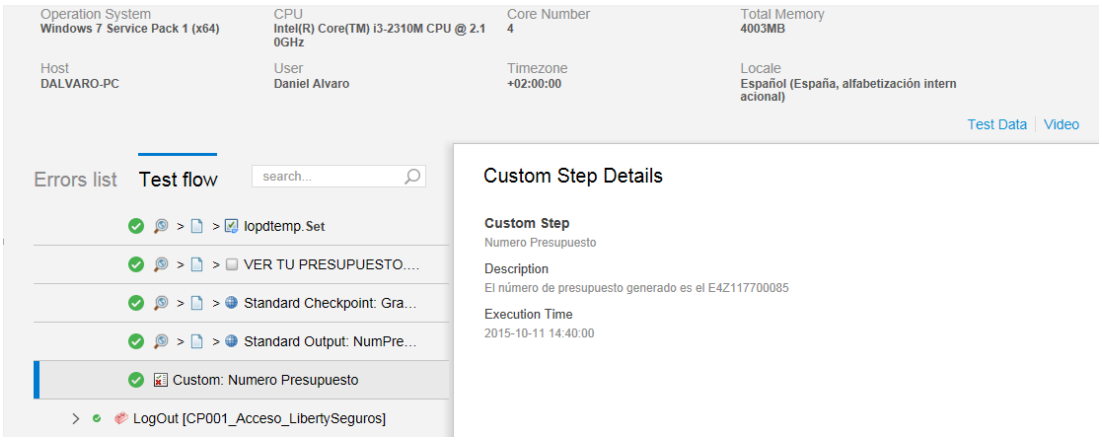


Figura 40 - Results Viewer

También se puede visualizar el “Analysis” en el caso de LoadRunner el cual contiene un sumario y diferentes gráficos con los resultados obtenidos, muy completo y que aporta gran información de la prueba efectuada.

Sin embargo, los complementos de análisis y reporte de los que disponen algunos de los productos software es demasiado pobre.

Selenium IDE no dispone, de manera individual, de un generador de informes de resultados. Por ello, la información que podemos sacar desde la interfaz es el número de casos ejecutados correctamente y el log, donde se visualizan los comandos correspondientes a los pasos ejecutados.

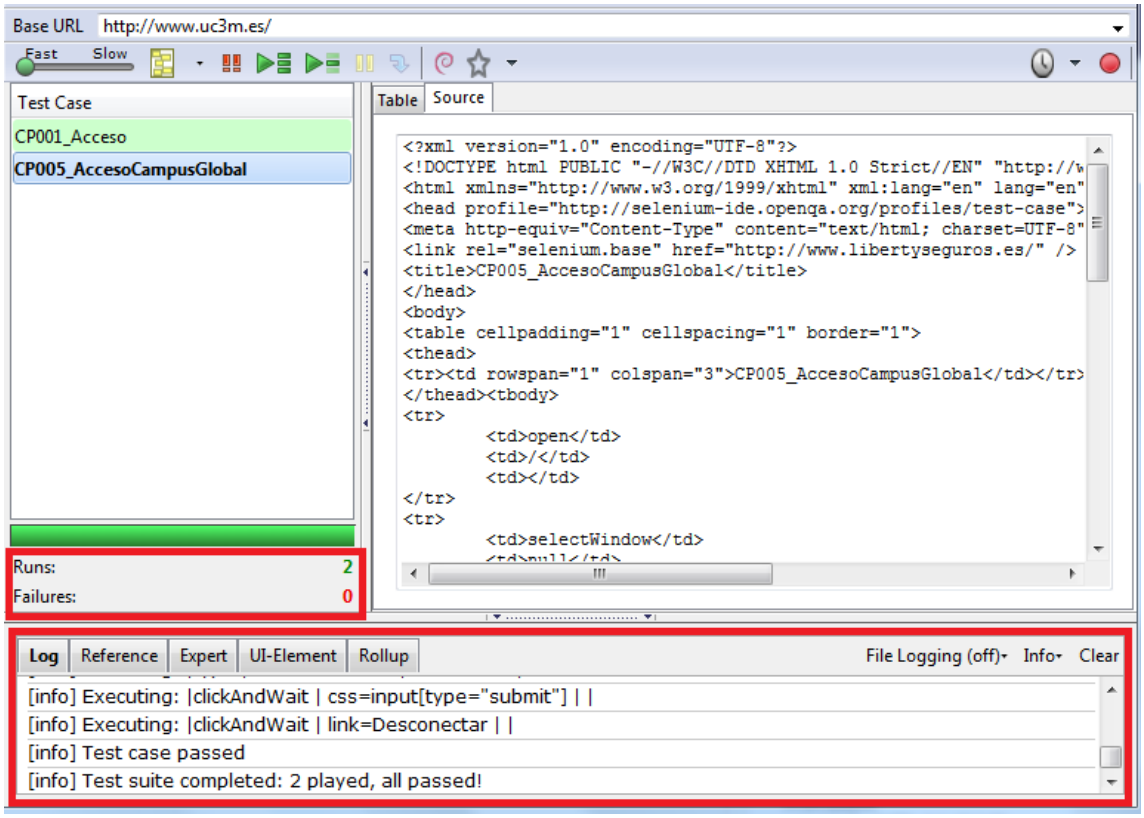


Figura 41 - Log Selenium IDE

En JMeter, al igual que en Selenium, los resultados no se pueden exportar a un formato útil y visualizarse fuera de la herramienta. Sin embargo, JMeter si permite configurar los informes de resultados a través de ficheros cvs y visualizarse de diversas maneras. A través de Jmeter se pueden generar árboles de ejecución, tablas de resultados, gráficos etc.

4.2 Pruebas funcionales automatizadas.

Se realizarán una serie de pruebas, concretamente 4 para la página de Liberty seguros y 4 para la página de la UC3M:

- Del CP001 al CP004 son los casos que se centran en la página de Liberty Seguro. Estos casos son una muestra de los casos típicos que formarían parte de las pruebas de regresión dentro del testing de una línea de negocio, ya que contienen funcionalidades básicas y que deben de ser probadas siempre antes de realizar una nueva subida de versión a producción.

- Del CP005 al CP008 el objetivo de estudio es la página web de la Universidad Carlos III de Madrid. En este caso la regresión está conformada por diversos casos que pasan por diferentes plataformas y/o arquitecturas, para comprobar la capacidad de las herramientas ante estas coyunturas.

- **Diseño:**

CP001	**Acceso Liberty**
Tipo Prueba	Prueba Regresión
Descripción	Comprobar el acceso a la página web de Liberty Seguros
Dependencias	No Aplica
Pasos	1. Acceder a la url 2. Comprobar que aparece correctamente
Datos	http://www.libertyseguros.es/
Resultado esperado	La página se muestra correctamente
Criterio de fallo	La página no se carga de manera adecuada
Aplicación/es	HP UFT, Selenium

Tabla 49 - Prueba automatizada CP001

CP002	Calcular presupuesto seguro coche
Tipo Prueba	Prueba Regresión
Descripción	Calcular un presupuesto de particulares de seguro de coche
Dependencias	CP001
Pasos	1. Acceder a la url 2. Ir a PARTICULARES > Seguros de coche 3. Calcula tu presupuesto 4. Rellenar datos obligatorios 5. Comprobar que se calcula el presupuesto
Datos	Datos personales, del seguro y del automóvil
Resultado esperado	Se genera un número de presupuesto
Criterio de fallo	No se calcula correctamente el presupuesto
Aplicación/es	HP UFT, Selenium

Tabla 50 - Prueba automatizada CP002

CP003	Encuentra tu mediador
Tipo Prueba	Prueba Regresión
Descripción	Buscar el mediador más cercano al lugar que le marcamos.
Dependencias	CP001
Pasos	1. Acceder a la url 2. Acceder a ENCUENTRA TU MEDIDADOR 3. Introducir código postal y Buscar 4. Comprobar que aparece algún resultado
Datos	http://www.libertyseguros.es/
Resultado esperado	Muestra el agente más cercano
Criterio de fallo	No muestra el agente más cercano
Aplicación/es	HP UFT, Selenium

Tabla 51 - Prueba automatizada CP003

CP004	Buscador de talleres
Tipo Prueba	Prueba Regresión
Descripción	Buscar el taller más cercano al lugar que le marcamos y mediante el mapa que aparece ver la ruta para llegar
Dependencias	CP001
Pasos	1. Acceder a la url 2. Acceder a Buscar un taller 3. Introducir tipo de taller, código postal y provincia y Buscar 4. Comprobar que aparece algún resultado 5. Seleccionar un taller del mapa y más info 6. Ver la ruta para llegar sobre el mapa
Datos	http://www.libertyseguros.es/,Tipo de taller, código postal y provincia
Resultado esperado	Muestra los talleres más cercanos
Criterio de fallo	No muestra talleres
Aplicación/es	HP UFT, Selenium

Tabla 52 - Prueba automatizada CP004

CP005	Acceder Campus Global
Tipo Prueba	Prueba Regresión
Descripción	Buscar el taller más cercano al lugar que le marcamos y mediante el mapa que aparece ver la ruta para llegar
Dependencias	No aplica
Pasos	1. Acceder a la url 2. Acceder al campus global y comprobar que se accede correctamente 3. Desconectar campus global y salir
Datos	www.uc3m.es, usuario y contraseña
Resultado esperado	Accedemos al campus global
Criterio de fallo	No accede al campus global
Aplicación/es	HP UFT, Selenium

Tabla 53 - Prueba automatizada CP005

CP006	Ver pdf horarios grado
Tipo Prueba	Prueba Regresión
Descripción	Acceder Grados y ver el calendario académico
Dependencias	No Aplica
Pasos	1. Acceder a la url 2. Acceder a ESTUDIOS > Estudia un Grado 3. Información Práctica > Calendarios académicos 4. Ver pdf calendario académico estudios de Grado y comprobar que aparece correctamente
Datos	www.uc3m.es
Resultado esperado	Ver pdf con el contenido del calendario de estudios de Grado
Criterio de fallo	No se muestra el pdf correctamente
Aplicación/es	HP UFT, Selenium

Tabla 54 - Prueba automatizada CP006

CP007	Ver vídeo UC3M digital
Tipo Prueba	Prueba Regresión
Descripción	Ver un vídeo a través de UC3M digital e interactuar con él
Dependencias	No Aplica
Pasos	1. Acceder a la url 2. Acceder a UC3M Digital > La UC3M en YouTube Edu 3. Pulsar enlace “Estudia en la UC3M” 4. Interactuar con el vídeo (pausar, ampliar …)
Datos	www.uc3m.es
Resultado esperado	Ver e interactuar con el vídeo a través del canal de youtube de UC3M
Criterio de fallo	No se muestra el vídeo correctamente
Aplicación/es	HP UFT, Selenium

Tabla 55 - Prueba automatizada CP007

CP008	Servicio correo PIC
Tipo Prueba	Prueba Regresión
Descripción	Buscar PIC Leganés y redactar correo
Dependencias	No Aplica
Pasos	1. Acceder a la url 2. Buscar PIC en el buscador general de la página de inicio 3. Seleccionar enlace Puntos de información de campus 4. Pulsar sobre el e-mail del PIC Leganés 5. Rellenar datos del correo y cerrar la ventana sin guardarlo
Datos	www.uc3m.es
Resultado esperado	Aparece la ventana del servidor de correo para enviar el e-mail al pic
Criterio de fallo	No se muestra el servidor de correo
Aplicación/es	HP UFT, Selenium

Tabla 56 - Prueba automatizada CP008

- **Implementación:**

El proceso de implementación de los casos de prueba requiere que las herramientas sean capaces de reconocer y trabajar con las plataformas software que forman parte de cada caso de prueba.

Por este motivo, inicialmente se mostrará el modo de implementación en las dos herramientas de automatización.

Para la implementación en HP UFT el lenguaje en que se generan los scripts es VBScript. El primer paso para conformarlo es se realiza una grabación de la navegación.

Figura 42 - Barra grabación HP UFT

Tras ello, dentro de la consola de desarrollo los scripts son moldeados, modificados y separados en acciones, conforme a las necesidades del caso de prueba.

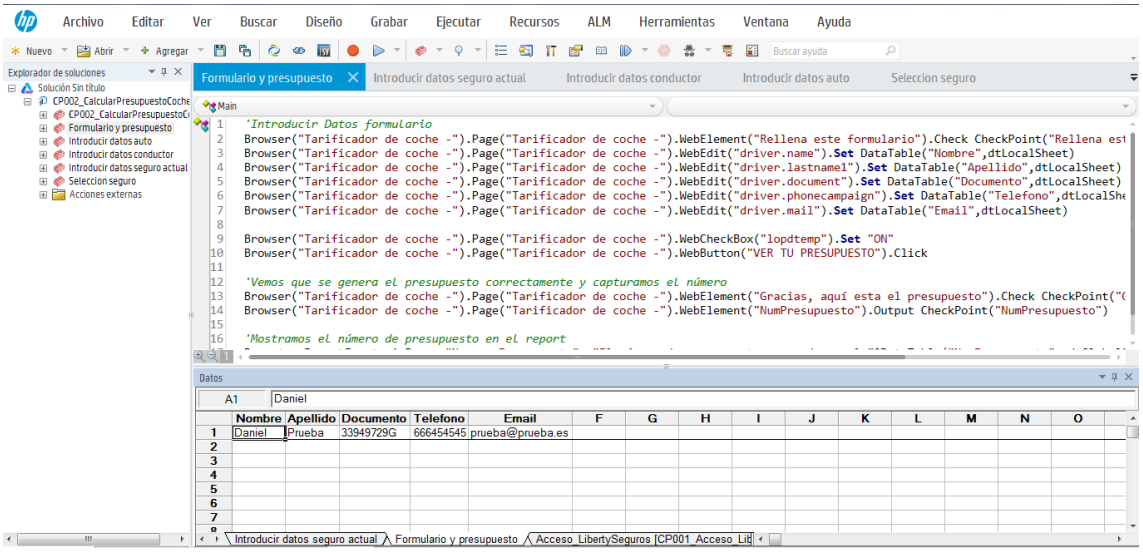


Figura 43 - Consola HP UFT

Por el contrario, Selenium IDE soporta variedad de lenguajes como ya se ha explicado en el punto de diferencias genéricas (Java, C#, PHP, etc.). En esta herramienta podemos tanto grabar/desarrollar en dos secciones diferentes.

La sección Table, donde la interfaz es más amena para el usuario, con ayuda de los comandos que podemos introducir.

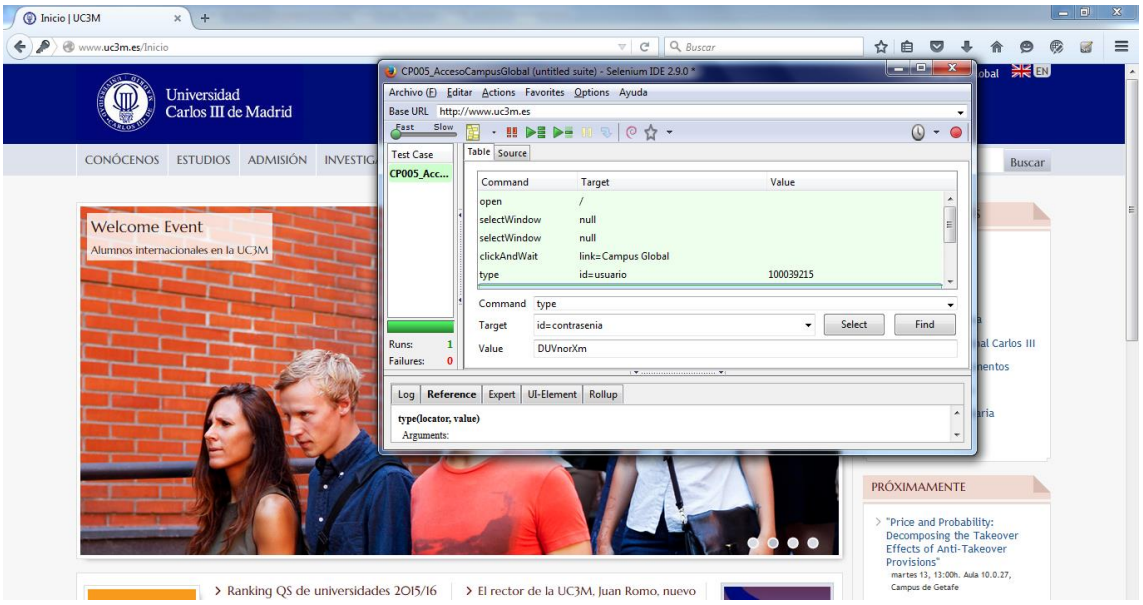


Figura 44 - Consola Table Selenium IDE

Por otro lado, aparece la parte Source que es donde se muestra la traducción en código de las interacciones llevadas a cabo. Esta parte requiere un mayor nivel de conocimiento de programación y familiaridad con el lenguaje utilizado.

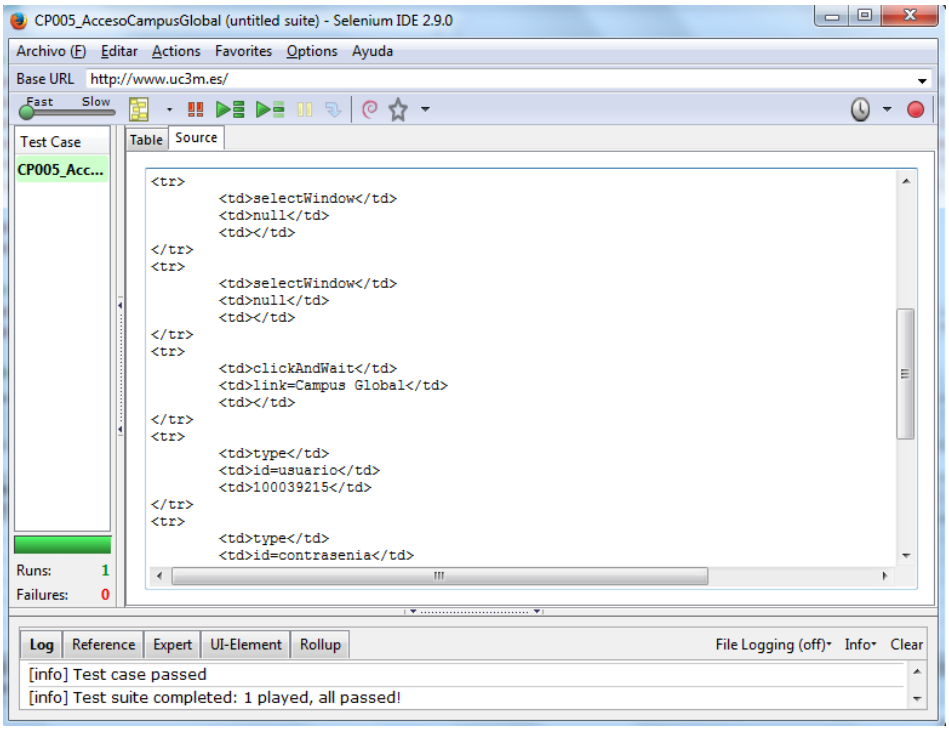


Figura 45 - Consola Source Selenium IDE

A continuación, se visualizará un cuadro en el que se indica si cada uno de los casos ha podido ser desarrollado correctamente.

Caso de prueba	HP UFT	Selenium IDE
CP001	✓	✓
CP002	✓	✓
CP003	✓	✓
CP004	✓	✗
CP005	✓	✓
CP006	✓	✗
CP007	✓	✓
CP008	✓	✗

Tabla 57 - Implementación CPs HP UFT y Selenium IDE

Los motivos por los que los casos de prueba CP004, CP006 y CP008 no han podido ser desarrollados por completo con la herramienta Selenium IDE, giran en torno a que este software sólo permite la grabación y reproducción en entornos basado en web.

Según esto, se detallan las carencias que han aparecido en el útil en cada uno de los casos de prueba:

- CP004: No es capaz de reconocer el mapa que se muestra con los talleres, para seleccionar uno y ver la ruta.

- CP006: No es capaz de trabajar ni con la ventana de aviso que nos dice si queremos abrir el documento o guardarlo; ni con el pdf.

- CP008: Al abrirse el Microsoft Outlook no reconoce esta ventana y no se puede interactuar con ella para completar los campos de asunto, cuerpo del correo etc.

Esto se debe a que HP UFT ofrece una amplia gama plataformas sobre las que grabar e interactuar. Permitiendo de este modo realizar diversos puntos de control y comprobaciones (checkpoints) de diferentes objetos y aspectos de la aplicación que se está testeando.

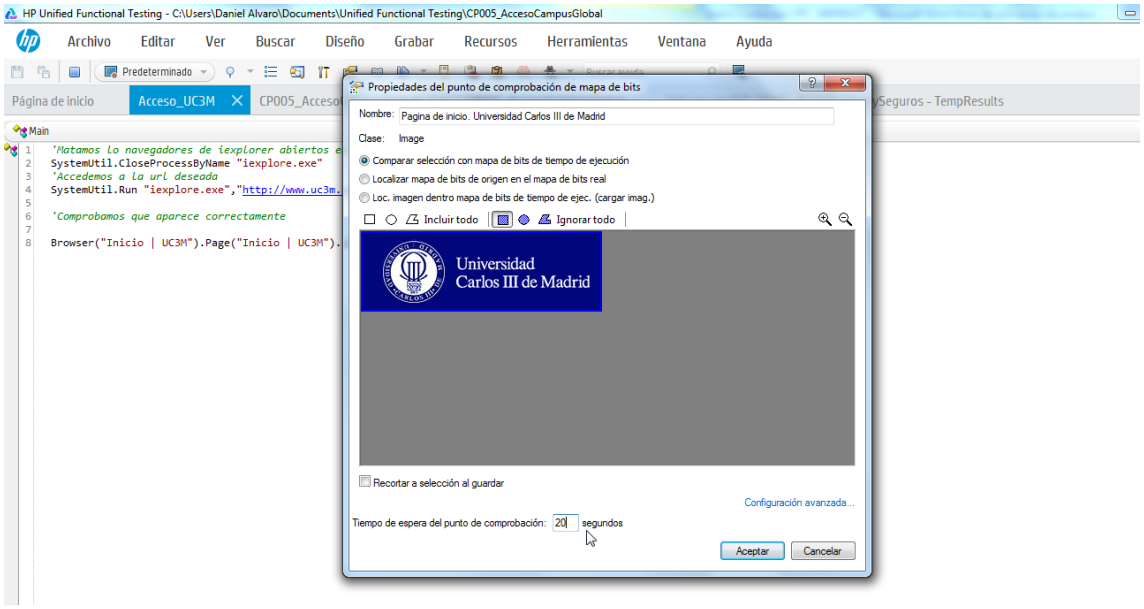


Figura 46 - Checkpoint Bitmap UFT

Además, también permite la grabación con el Insight Object, que se trata de una grabación de modo imagen y que es de gran ayuda siempre que nos encontremos con alguna tecnología no reconocida por el software.

Todos los casos de prueba, con sus scripts modelados en las diferentes soluciones software, se encuentran adjuntos en el cd que acompaña este documento.

- **Ejecución:**

En este punto, antes de realizar la ejecución se encapsulan y ordenan las acciones generadas para realizar la navegación deseada. Esto se ve más claramente gracias a la posibilidad que nos ofrece HP UFT de mostrar el flujo de las acciones en modo árbol.

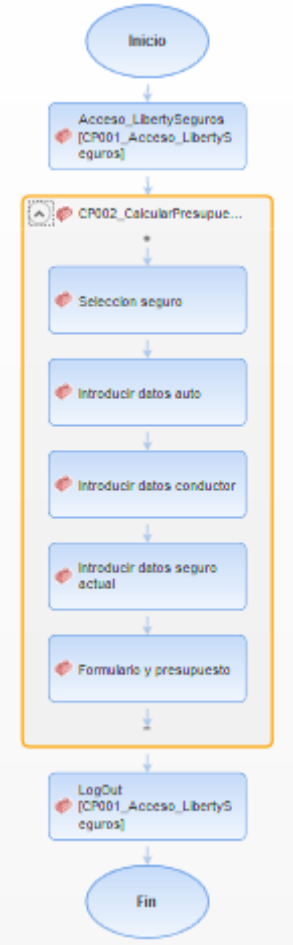


Figura 47 - Flujo de ejecución UFT

También definimos previamente el modo de ejecución. Dentro de las opciones de ejecución seleccionamos el modo rápido (Fast). Es decir, el script se ejecutará respetando las sincronizaciones y latencias de la web, pero no tendrá un retraso extra definido entre paso y paso.

En Selenium IDE, para ordenar la secuencia de las acciones y ver cómo se irán ejecutando podemos fijarnos en la sección de la izquierda llamada Test Case. En esta sección también marcamos el nivel de velocidad de la ejecución a Fast.

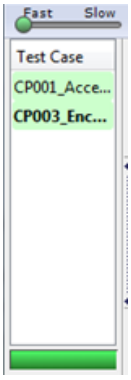


Figura 48 - Flujo de ejecución Selenium ID

Todos los vídeos de las pruebas ejecutadas con HP Unified Functional Testing, en formato estandarizado ".exe", se encuentran adjuntos en el cd adjunto del proyecto.

4.3 Pruebas de rendimiento automatizadas.

Se van a realizar un par de escenarios de pruebas en los que se incluirá un nuevo caso de prueba. Este caso no debe de ser muy extenso para que se pueda hacer más de una iteración durante el tiempo de la prueba. Se implementará la navegación del mismo caso de prueba en ambas herramientas y se estudiará cómo responden cada una de ellas en los diferentes escenarios en los que se ejecutará. De esta manera, podremos contrastar los distintos productos y su eficiencia, ya que ése es el objetivo de dichas pruebas.

- **Diseño:**

CP009	Conócenos Campus Leganés
Tipo Prueba	Prueba Regresión
Descripción	Acceder a la información del campus de Leganés
Dependencias	No Aplica
Pasos	1. Acceder a la url 2. Pulsar sobre CONÓCENOS 3. Seleccionar a Los Campus 4. Acceder a Campus de Leganés
Datos	http://www.uc3m.es/
Resultado esperado	La página principal del campus de Leganés se muestra correctamente
Criterio de fallo	La página no se carga de manera adecuada
Aplicación/es	HP LoadRunner, JMeter

Tabla 58 - Prueba regresión CP009

Se ha elegido éste, porque sería una prueba que formaría parte de las pruebas de regresión y a su vez es un modelo de prueba de rendimiento, cuyas llamadas al servidor de la página principal son del tipo de prueba de carga más demandadas y estudiadas cuando se desea probar el rendimiento de un producto web.

Los escenarios que se van a abordar, se realizarán después de que se haya implementado el caso de prueba en cada producto software. Sin embargo, voy a mostrar el diseño que va a tener cada uno de los escenarios de pruebas que implementaremos y utilizaremos para la ejecución de las pruebas.

Para diferenciar los dos escenarios, ya que se van a referir al mismo caso de prueba, se variará la carga, el modo en que se introduce, el tramo de estabilidad y el modo en que va liberándose.

EP01	5Vusers
Tipo Prueba	Escenario de Pruebas
Descripción	Escenario de Prueba 1
Caso de prueba	CP009
Vusers	5 usuarios
Rampa de Entrada	Entrada simultánea
Tramo estabilidad	1 minuto
Rampa de Salida	Salida simultánea
Aplicación/es	HP LoadRunner, Jmeter

Tabla 59 - Escenario de prueba EP01

EP02	10Vusers
Tipo Prueba	Escenario de Pruebas
Descripción	Escenario de Prueba 2
Caso de prueba	CP009
Vusers	10 usuarios
Rampa de Entrada	1 usuario cada segundo
Tramo estabilidad	30 segundos
Rampa de Salida	1 usuarios cada segundo
Aplicación/es	HP LoadRunner, Jmeter

Tabla 60 - Escenario de prueba EP02

Las pruebas de rendimiento que se realizan en un ámbito de negocio real y que tienen como objeto de estudio páginas web, requieren una gran carga para comprobar el número de usuarios concurrentes que soporta, determinar el punto de máximo rendimiento etc. Gracias a este tipo de testing se podrán tomar medidas para que la plataforma web no sufra caídas o colapsos cuando tiene mucha demanda de usuarios o carga de servidores. Esto impediría dar el servicio deseado, provocando pérdida de credibilidad en el usuario, de nuevos clientes potenciales, de dinero etc.

Por ello, en nuestros escenarios hay que tener en cuenta no utilizar una carga muy alta ya que se tratan de pruebas en un entorno real y no se quieren colapsar los servidores de las mismas. Además la infraestructura de la que disponemos no es la más adecuada ya que requeriríamos generadores de carga y servidores ajenos para no saturarnos.

- **Implementación:**

A la hora de llevar a desarrollo las pruebas de carga planteadas anteriormente, en primer lugar se realiza la grabación de la navegación del CP009 en las dos plataformas de pruebas software que disponemos, Apache JMeter y HP LoadRunner. Tras ello, se crearán los escenarios donde se incluirá dicho caso de prueba y se configurará el modo en que se tratará la carga.

Para llevar a cabo la grabación en Jmeter debemos de configurar tanto la herramienta como el navegador que utilicemos, IExplorer, Mozilla etc. Yo he utilizado Mozilla Firefox porque cuando se realice la implementación en HPLoadRunner se hará con el protocolo TruClient, y éste también se usa Mozilla.

En JMeter se añade un Grupo de Hilos al Plan de Pruebas, aquí es donde aparecerán todas las peticiones que se realicen durante la grabación de la navegación. Tras ello se añade en el banco de trabajo el Servidor Proxy HTTP, y se configura conforme a las pruebas que vamos a realizar completando los campos Puerto, HTTPS Domains, etc.

En este apartado también se he rellenado el campo “URL Patrones a Excluir”, que es donde se ponen los tipos de HTTP Request no relevantes y que se quieren excluir de la grabación. La expresión regular que he introducido en este punto es: .*\.(gif|png|jpg||css|js)

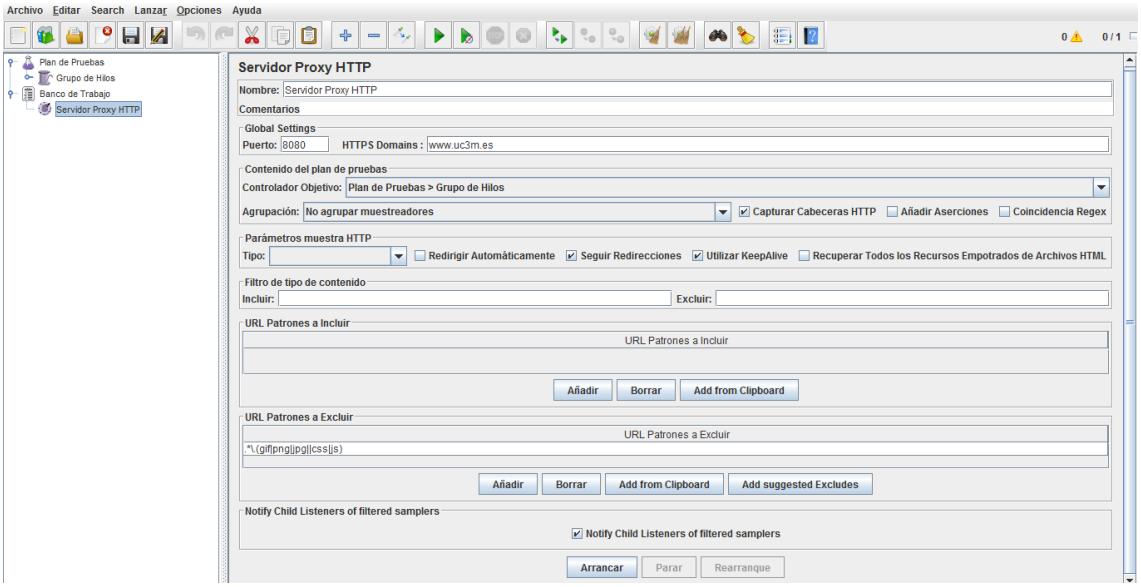


Figura 49 - Servidor Proxy HTTP JMeter

Ahora configuro el explorador que vamos a manejar para poder realizar la navegación. Para ello utilizaremos la configuración del proxy con Address *localhost* y Puerto *8080*.

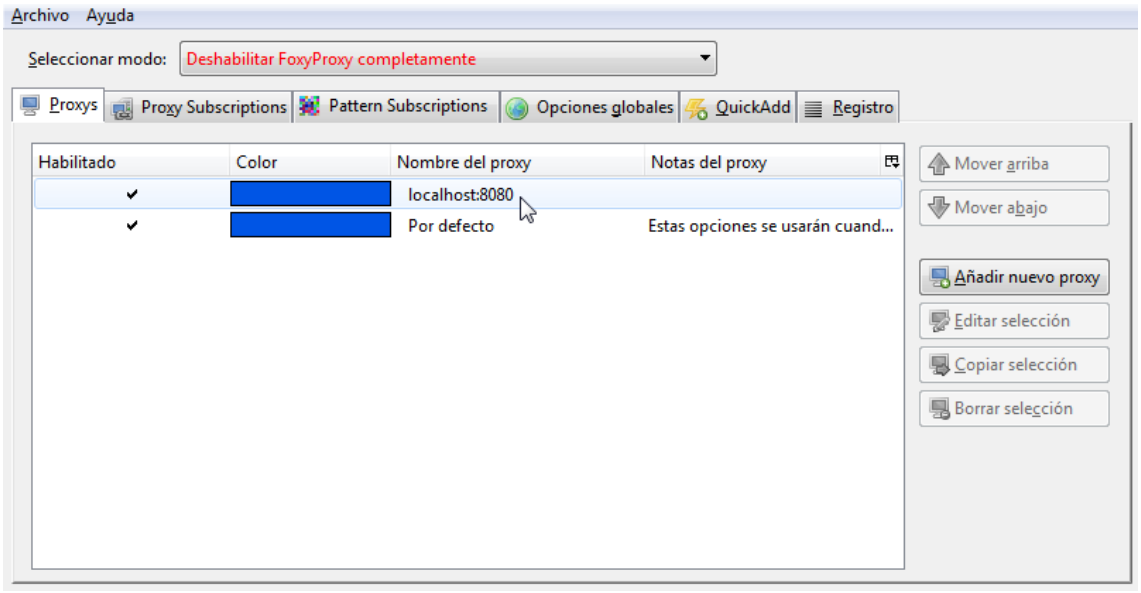


Figura 50 - Configuración Proxy Navegador

El próximo paso ya es pulsar el botón arrancar y realizar todos los pasos sobre el navegador web hasta llegar al último paso del mismo, generando un número de peticiones HTTP que aparecen conjuntamente en el grupo de hilos. Esta secuencia será la que ejecutaremos después.

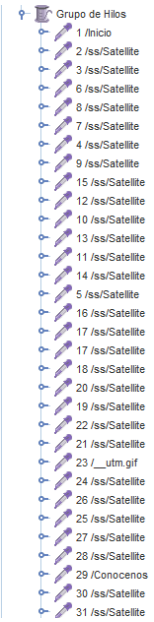


Figura 51 - Configuración Proxy Navegador

En HP LoadRunner realizamos la grabación del CP009 a través del VuGen con protocolo TruClient. Este protocolo es uno de los más actuales de esta solución y es más cercano para el usuario, tanto para la grabación como para la depuración y modificación del mismo.

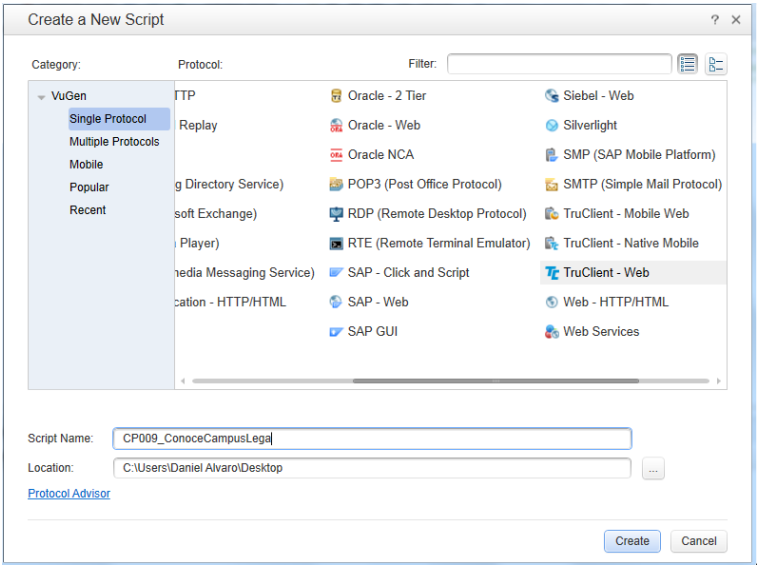


Figura 52 - New Script TruClient LoadRunner

En el VuGen se pulsa el botón de Develop Script y entra en la interfaz del TruClient. Aquí se realiza la grabación de la navegación del CP sobre Mozilla Firefox y las peticiones correspondientes a las acciones simuladas se engloban en transacciones. Esto permite que cuando se realice tanto la ejecución como el análisis, se localicen e identifiquen más rápidamente las transacciones realizadas.

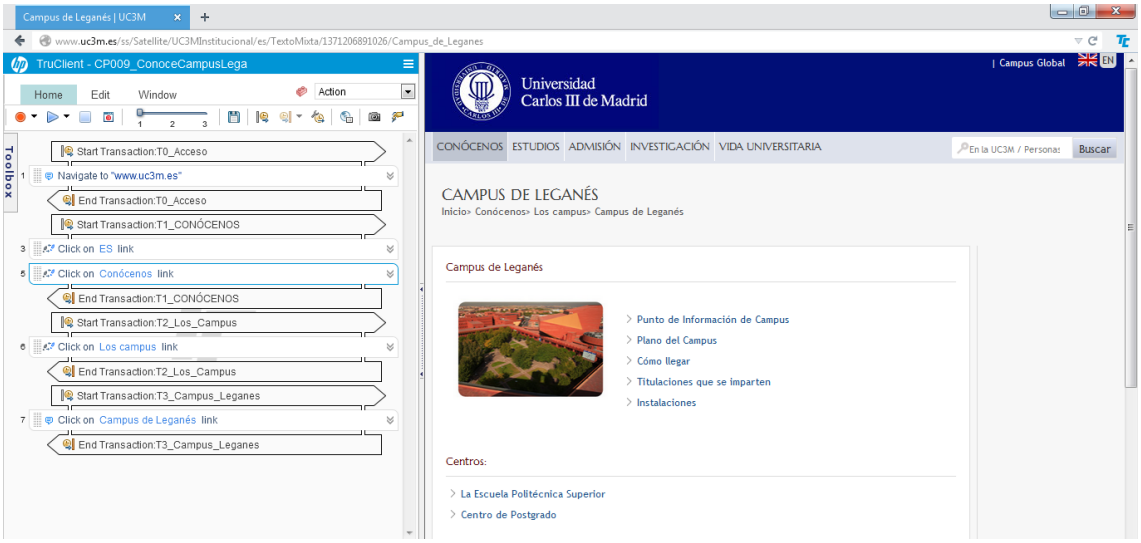


Figura 53 - Grabación TruClient LoadRunner

Tras grabar el caso de prueba en ambas herramientas, se implementan los escenarios de prueba.

En JMeter la configuración de los escenarios se efectúa en el Grupo de Hilos. Dentro de este grupo también se agregan los reportes y gráficos que se desean visualizar tanto en tiempo de ejecución como al finalizar.

Se realizará el EP01 que consiste en 5 hilos que entraran sin latencia entre ellos y que se ejecutarán durante 60 segundos.

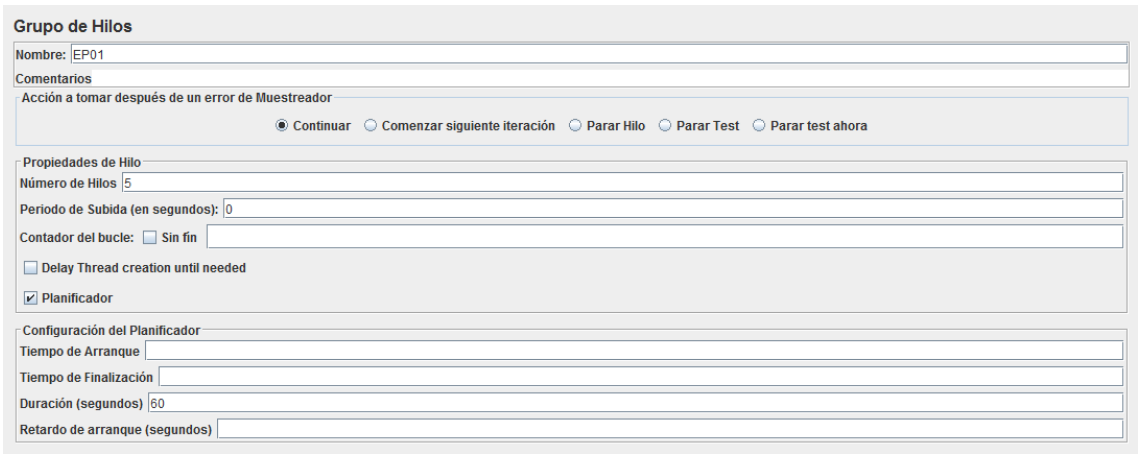


Figura 54 - JMeter EP01

El EP02 trata 10 hilos que entraran con 1 segundo de diferencia entre ellos y se ejecutarán durante 30 segundos.

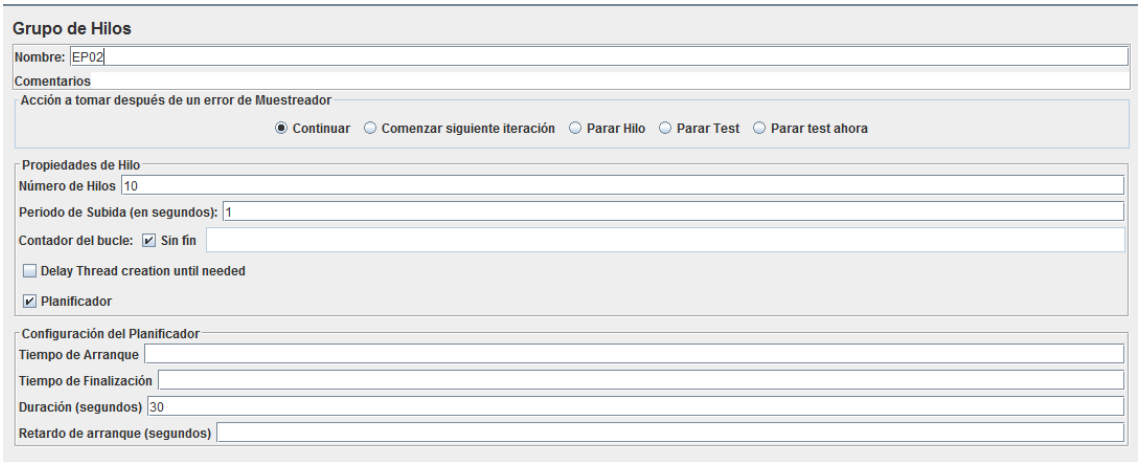


Figura 55 - JMeter EP02

Para implementar los escenarios en la herramienta de HP utilizamos el Controller. Este módulo permite seleccionar los VUser (usuarios virtuales), la rampa de entrada, el tiempo de ejecución, la rampa de salida etc., de una manera más visual. Además se cargará el CP009 grabado previamente al escenario de ejecución.

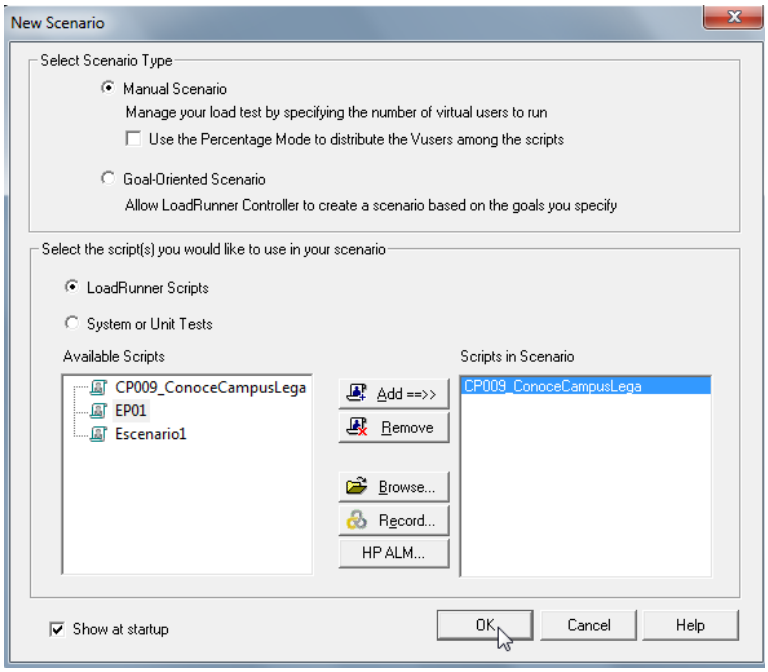


Figura 56 - Nuevo Escenario HP Loadrunner

El escenario EP01 tiene 5 VUsers que entran de manera simultánea, se mantendrán durante 1 minuto y salen simultáneamente.

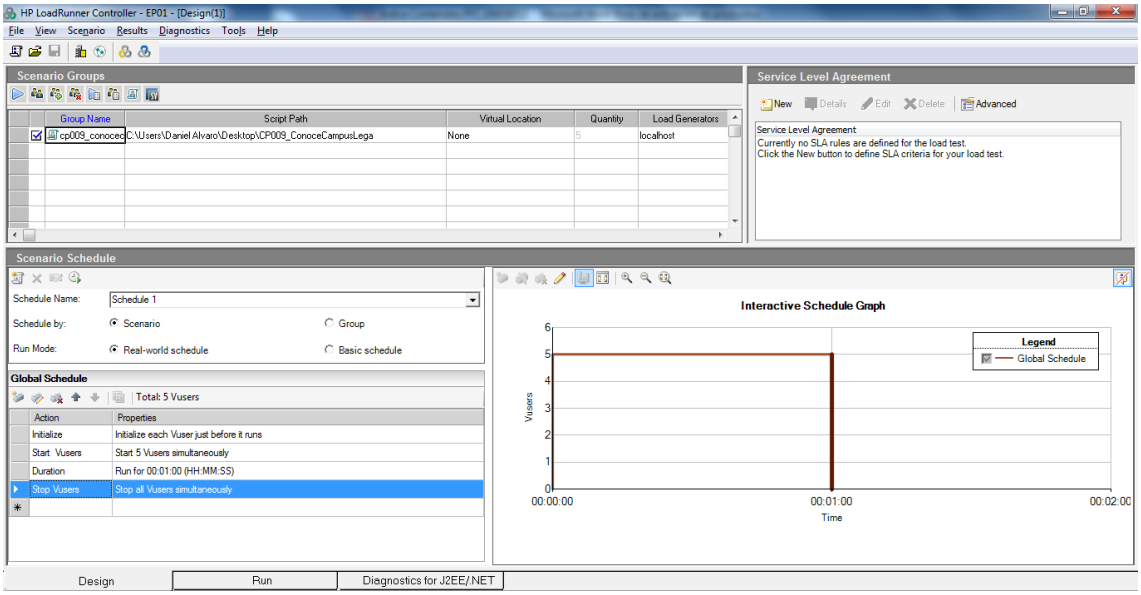


Figura 57 - EP01 Controller

El escenario EP02 tiene 10 VUsers que entran 1 cada segundo, se mantendrán durante 30 segundos y salen paulatinamente 1 por segundo.

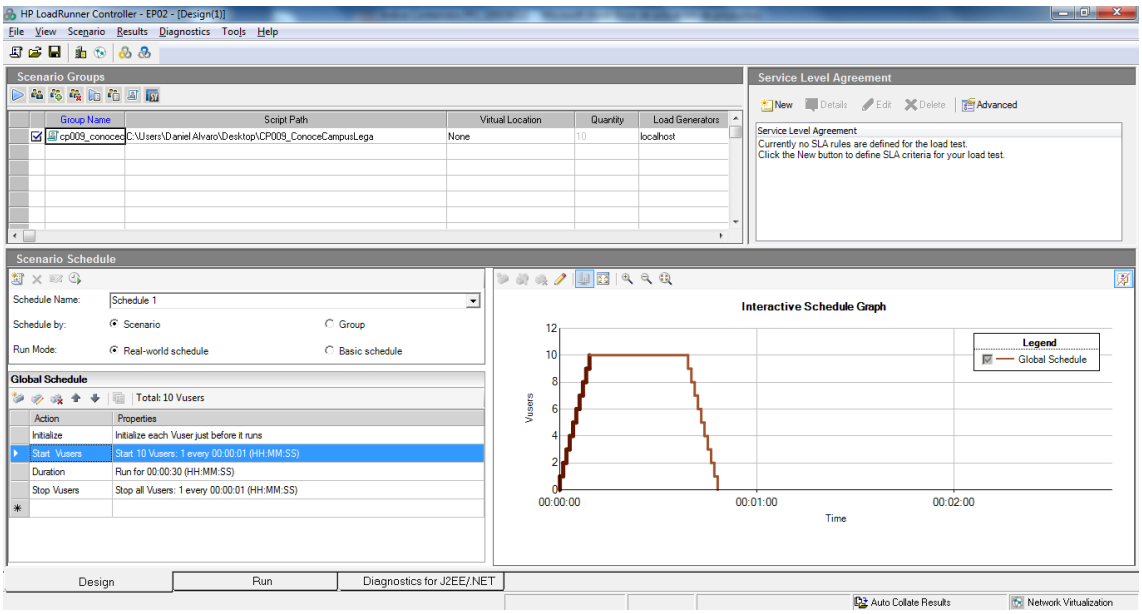


Figura 58 - EP02 Controller

Los archivos correspondientes al caso de prueba CP009 y los dos escenarios, desarrollados en cada una de las herramientas, se anexan en el CD adjunto al proyecto.

- **Ejecución:**

La ejecución de los escenarios provoca un envío constante de peticiones y de respuestas por parte del servidor con cada iteración que se realiza.

En JMeter cuando se ejecuta el EP se observa como las peticiones se van enviando, los tiempos de respuesta, si se han completado correctamente etc. Estos valores se pueden visualizar en formato resumen, de árbol, en modo gráfico etc., según se haya configurado.

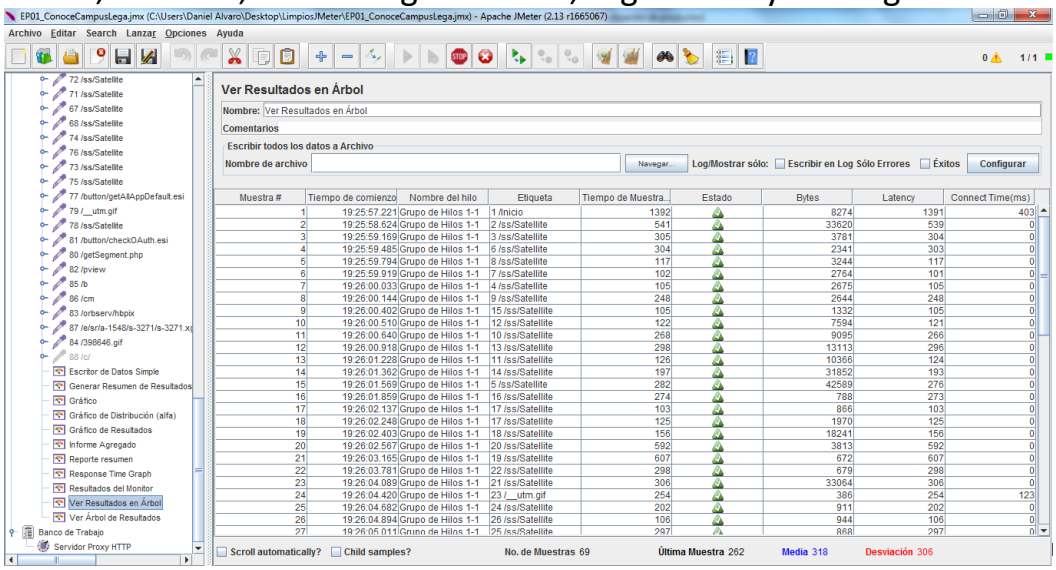


Figura 59 - Ejecución JMeter

En HP LoadRunner se puede observar en tiempo de ejecución un resumen de los usuarios virtuales que está corriendo, las transacciones pasadas y las fallidas, los errores que han aparecido etc. También podemos configurar y ver diferentes gráficas durante la prueba.

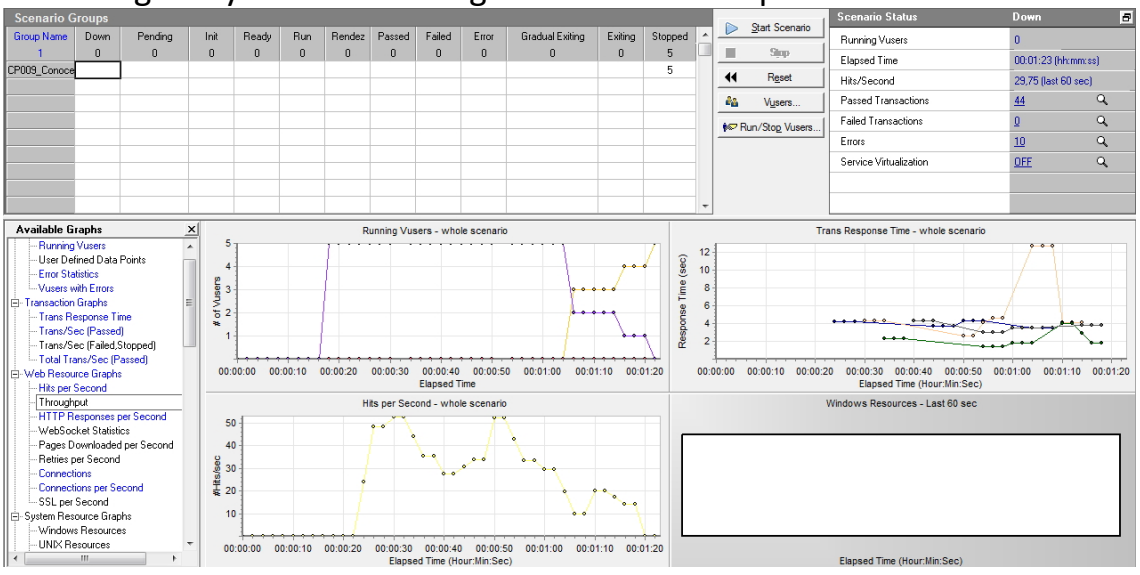


Figura 60 - Ejecución HP LoadRunner

4.4 Matriz de trazabilidad

Para resumir los objetivos y requisitos alcanzados con las pruebas se realiza una matriz de trazabilidad. Esta matriz tiene en su eje vertical el conjunto de identificadores de los casos de pruebas y escenarios realizados, y en su eje horizontal, los identificadores de requisitos software.

	SR_RF_01	SR_NF_02	SR_RF_03	SR_RF_04	SR_NF_05	SR_NF_06	SR_NF_07	SR_RF_08	SR_NF_09	SR_NF_10	SR_RF_11	SR_RF_12	SR_RF_13	SR_RF_14	SR_NF_15	SR_NF_16
PF001					X											
PF002	X			X				X								X
PF003				X												
PF004		X							X							
PF005	X		X			X	X		X	X	X					
PF006		X						X	X	X				X	X	X
PF007												X	X	X		
CP001	X		X	X				X			X	X	X	X		
CP002	X		X	X	X			X	X		X	X	X	X		
CP003	X		X	X				X			X	X	X	X		
CP004	X		X	X				X	X	X	X	X	X	X		
CP005	X		X	X			X	X			X	X	X	X		
CP006	X		X	X				X		X	X	X	X	X		
CP007	X		X	X				X			X	X	X	X		
CP008	X		X	X				X	X	X	X	X	X	X		
CP009	X		X	X							X	X	X	X		
EP01		X	X	X	X	X					X	X	X	X	X	X
EP02		X	X	X	X	X					X	X	X	X	X	X

Tabla 61 - Matriz trazabilidad casos de prueba y req. técnicos

Capítulo 5

Evaluación de los resultados y comparativa.

En este punto se muestra el tema central de todo este estudio, es decir la evaluación de los resultados obtenidos en las pruebas previamente realizaras

Tras ellos se hará una comparación entre las herramientas, tanto de manera cualitativa como cuantitativa, atendiendo a las conclusiones producto del estudio.

5.1 Análisis y Evaluación de los resultados.

5.1.1 Pruebas manuales.

Para analizar las pruebas manuales se dará la puntuación explicada para las pruebas cualitativas, que indicará el grado de satisfacción al realizar cada prueba con cada una de las herramientas.

	HP UFT	HP LoadRunner	Selenium	JMeter
PF001	2	2	4	3
PF002	5	5	3	2
PF003	4	4	2	3
PF004	2	2	4	4
PF005	5	4	3	2
PF006	4	4	4	4
PF007	5	5	2	3
TOTAL	27	26	22	21

Tabla 62 - Resultado pruebas manuales

En la parte de la instalación (PF001) las herramientas comerciales son las menos puntuadas ya que son las que más permisos y recursos requieren, además de licencias y registrarse en la plataforma. Mientras que las gratuitas son más rápidas y necesitan menos requisitos de sistema.

La interfaz de usuario de las herramientas de pago son mucho más potentes y más amigables para el usuario, mientras que las otras necesitan un pequeño salto de calidad para hacerlas más manejables.

Todas las herramientas tienen una sección de ayuda y apoyo al usuario. Pero dentro de este nivel observamos claras diferencias, como que en Selenium IDE esta ayuda es un re direccionamiento on-line a la página de Selenium. JMeter tiene bastantes artículos pero es una ayuda muy estática, mientras que las herramientas de HP tienen la posibilidad tanto de navegar por diferentes secciones, buscar por términos, etc. A parte, las suites de HP son las únicas que tienen servicio de soporte por parte de la entidad propietaria del software.

La integración con otras herramientas es mucho más potente en las libres y gratuitas ya que el código que las conforma está en continuo desarrollo por diferentes programadores. Esto permite que se pueda innovar e integrar con otro software adecuándolo al mismo. Por el contrario, los productos de pago tienen la posibilidad de integrarlo con otros productos pero del mismo dueño, y por consiguiente de pago también. Pero éstos nos aportan un mayor nivel de confianza y estabilidad pues siguen un proceso de calidad claramente marcado.

La grabación es mucho más extensa y con mucha más variedad de posibilidades en las herramientas de pago. En las gratuitas está más limitada la diversidad de arquitecturas de las que tiene cobertura. La reutilización y encapsulación de acciones y funcionalidades es mucho más ágil y mantenible tanto en HP UFT como en HP LoadRunner. En todas las herramientas objeto de estudio se puede depurar el código y utilizar útiles como puntos de ruptura para ayudarnos a corregir los problemas encontrados.

La ejecución de los casos de prueba en las diferentes herramientas es bastante sencilla en todas ellas y si se ha implementado correctamente el resultado será óptimo.

Por último, en la parte relativa a los módulos que recogen los resultados y analizan la situación de la ejecución, está más avanzado y es más completo y transparente en el software privativo. Ya que nos da más información y capacidad para poder utilizar esa documentación fuera del útil. Sin embargo, las gratuitas además de darnos menor nivel de detalle de la secuencia ejecutada tampoco nos permite trabajar con los informes que podamos generar fuera del producto. E incluso en el caso de Selenium IDE, llegar a necesitar otro software para poder generar informes ya que por sí sólo no es capaz de generarlos.

5.1.2 Pruebas funcionales automatizadas.

Como ya se ha explicado en los primeros puntos del estudio, el tiempo que se tarda en ejecutar las pruebas funcionales automatizadas no es uno de los objetivos principales de la automatización de pruebas funcionales, sino que su finalidad es, entre otras, liberar recursos y utilizarlos en otras labores, ejecutar de manera desatendida, mantener niveles de objetividad intactos, etc.

Sin embargo, para aportar otro punto de análisis y diferenciación entre las herramientas automáticas, se ejecutan los casos de prueba automatizados y se toma la duración de ejecución de cada una de las pruebas implementadas. Estos tiempos se especifican en segundos y se muestran en el siguiente cuadro.

Caso de prueba	Tiempo ejecución HP UFT (s)	Tiempo ejecución Selenium IDE (s)
CP001	15	8
CP002	62	34
CP003	43	14
CP004	42	✘
CP005	34	20
CP006	57	✘
CP007	50	16
CP008	36	✘

Tabla 63 - Tiempos ejecución HP UFT y Selenium IDE

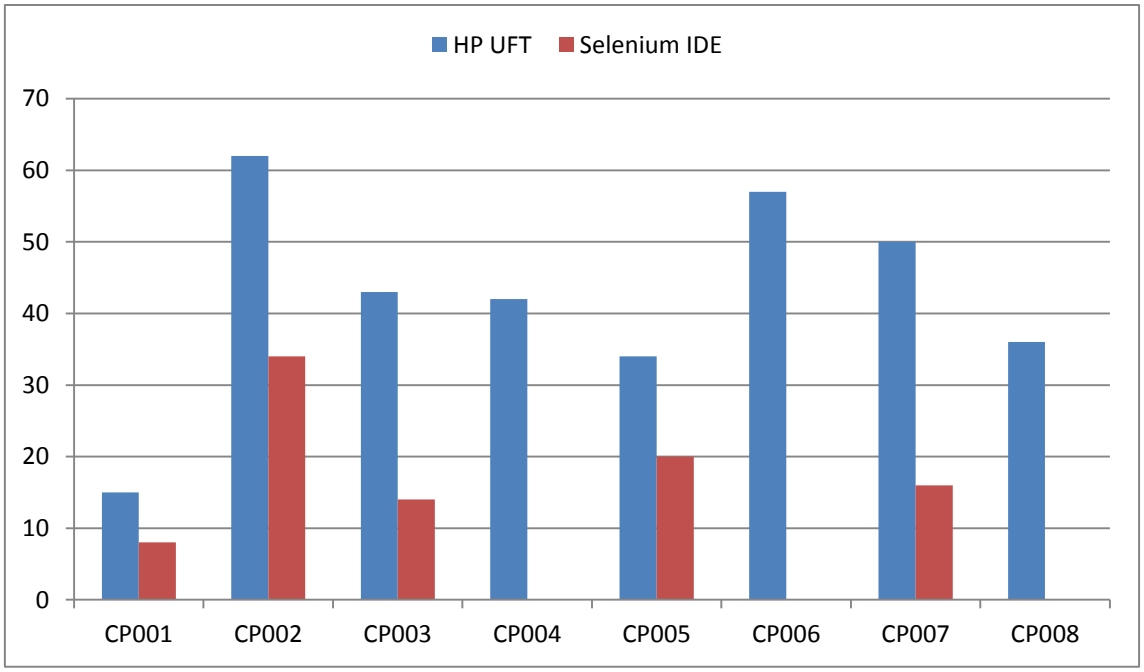


Figura 61 - Tiempos ejecución HP UFT y Selenium IDE

Como se observa en la tabla los tiempos de ejecución de UFT son siempre bastante más altos que los de Selenium IDE. Esto se debe a varios motivos:

- El browser de Mozilla Firefox siempre está abierto previamente a empezar la prueba con Selenium IDE, ya que se trata de un plugin de este navegador y lo necesita. Sin embargo, las pruebas de HP UFT cierran y abren el explorador IExplorer 11, con el tiempo que supone la carga del mismo.

- Los casos de la herramienta comercial tienen un mayor número de checkpoints y comprobaciones, ya que esta herramienta nos lo permite.

- HP UFT nos permite realizar vídeos de la ejecución (módulo Screen Recorder), lo cual también incurre en que tarde algo más en generar el resultado de la prueba.

- La calidad y cantidad de resultados recogidos por la herramienta de pago es más extenso y completo que el de Selenium IDE. Por ello la recolección de resultados al finalizar la prueba es más lento en la solución de HP.

La memoria y la CPU que consumen tanto el software libre como el software privado son muy parecidos. Ya que, aunque el plugin de Selenium IDE por separado no consume apenas recursos, como está asociado al navegador Firefox, éste sí consume más requisitos del sistema. Por el contrario, HP UFT requiere más memoria y CPU que Selenium IDE individualmente, pero al unir cada uno con su navegador la media es muy similar.

La siguiente tabla muestra aproximadamente una media de tanto la memoria como de la CPU consumida por cada herramienta, en estado de inactividad y en ejecución.

CONSUMOS EN ESTADO DE INACTIVIDAD		
Herramienta + Explorador	**Memoria (KB)**	**CPU (%)**
HP UFT + IE11	268.000	1-2
Selenium IDE + Mozilla Firefox	256.000	1-2

Tabla 64 - Consumos inactividad

CONSUMOS EN ESTADO DE ACTIVIDAD		
Herramienta + Explorador	**Memoria (KB)**	**CPU (%)**
HP UFT + IE11	284.000	9-12
Selenium IDE + Mozilla Firefox	293.000	13

Tabla 65 - Consumos actividad

5.1.3 Pruebas de rendimiento automatizadas.

Tras finalizar los escenarios de prueba podemos observar los resultados recogidos a través de los gráficos y diferentes receptores establecidos en el escenario. En JMeter, todos los recursos que dispone para ver muestras, resúmenes etc., se encuentran integrados en la misma interfaz y su análisis requiere un tratamiento de los datos.

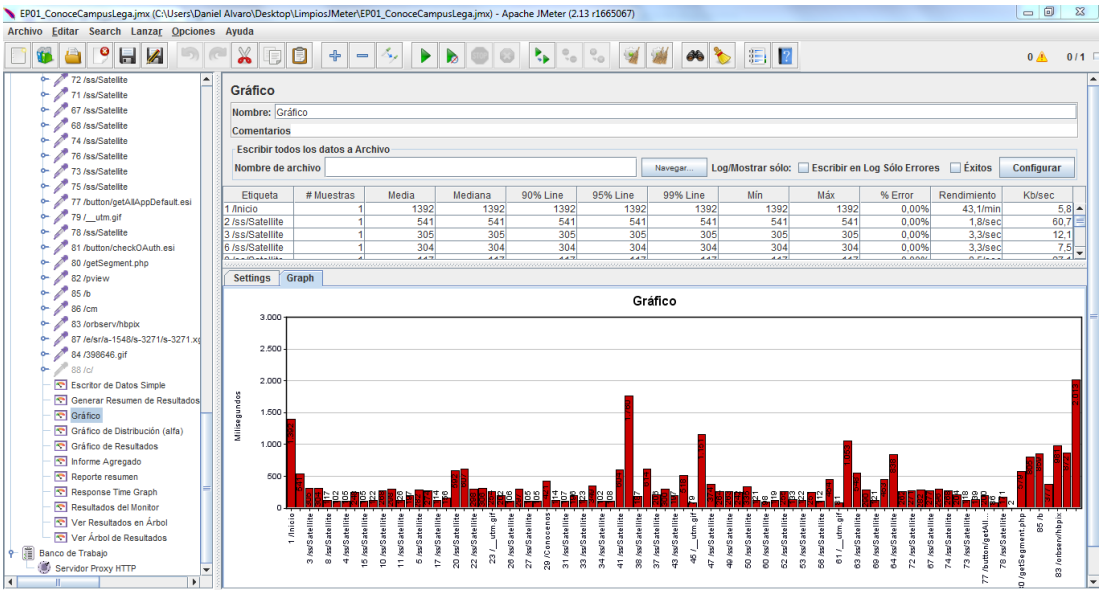


Figura 62 - Gráfico JMeter

En HP Loadrunner además de poder visualizar los resultados en el Controller, tenemos la opción de que tras hacer realizarse un Collate de los datos recogidos se genere un fichero de resultados que se trata por el módulo llamado Analysis. Esta parte permite configurar las plantillas y gráficas a mostrar: Throughput, Hits por segundo, tiempos respuesta etc.

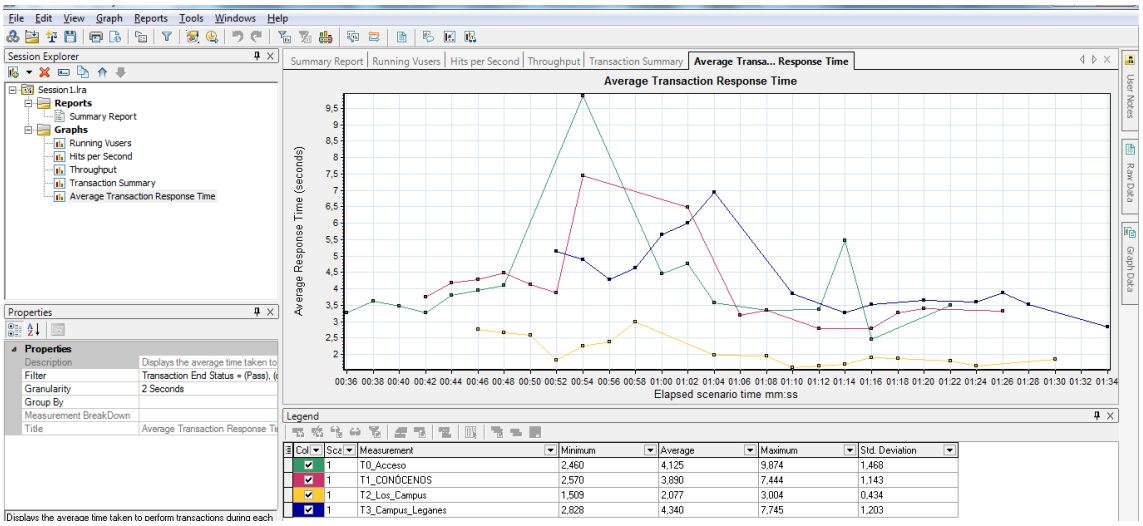


Figura 63 - Analysis Loadrunner

Como en LoadRunner se pueden unificar las peticiones en transacciones, esto permite ver y unificar más claramente qué datos de los obtenidos se corresponden a las operaciones realizadas.

En el Summary Report se puede observar información relativa a la prueba: Transacciones totales, tiempos de respuesta, desviación estándar etc.

Transactions: Total Passed: 44 Total Failed: 0 Total Stopped: 0 Average Response Time

Transaction Name	SLA Status	Minimum	Average	Maximum	Std. Deviation	90 Percent	Pass	Fail	Stop
T0 Acceso		3,349	4,018	4,885	0,418	4,271	11	0	0
T1 CONÓCENOS		2,602	4,987	12,674	2,501	4,968	11	0	0
T2 Los Campus		1,363	2,137	3,955	0,696	2,473	11	0	0
T3 Campus Leganes		3,001	3,905	6,48	0,883	4,117	11	0	0

Figura 64 - Summary Report EP01

Transactions: Total Passed: 84 Total Failed: 0 Total Stopped: 0 Average Response Time

Transaction Name	SLA Status	Minimum	Average	Maximum	Std. Deviation	90 Percent	Pass	Fail	Stop
T0 Acceso		2,46	4,125	9,874	1,468	5,054	21	0	0
T1 CONÓCENOS		2,57	3,89	7,444	1,143	4,493	21	0	0
T2 Los Campus		1,509	2,077	3,004	0,434	2,7	21	0	0
T3 Campus Leganes		2,828	4,34	7,745	1,203	6,012	21	0	0

Figura 65 - Summary Report EP02

JMeter en estado de inactividad supone un consumo de memoria de apenas 38.500 KB. Pero cuando se ejecuta un escenario de prueba, aumentan los recursos que consume en relación de la carga de hilos concurrentes que haga.

En estado de reposo, los recursos consumidos por los módulos de HP (VUgen, Controller y Analysis) no son demasiado grandes, por ejemplo la memoria varía alrededor de 41.500 KB. Pero cuando se realiza una grabación en VuGen o una ejecución en Controller, el consumo aumenta.

El mayor consumo se hace cuando se ejecuta un escenario de pruebas, ya que el Controller abre tantos procesos de navegador como VUsers hay corriendo al mismo tiempo y se puede llegar a colapsar nuestra máquina. Por eso, lo idóneo para utilizar este tipo de herramientas y sacarles todo el rendimiento, es tener una infraestructura de máquinas que den soporte y podamos utilizar de inyectores de carga.

Por ejemplo, en el EP02 a mitad de la prueba el Controller dio aviso de aparición de errores debido a que se había sobrepasado el 80 % de la CPU del equipo.

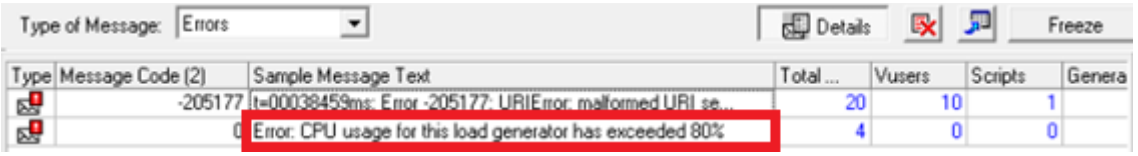


Figura 66 - Error CPU EP02

5.2 Comparativa entre herramientas.

5.2.1 Análisis cualitativo.

Para la comparativa cualitativa de las herramientas, además de apoyarnos en todo lo que hemos visto a lo largo del proyecto, se realiza una encuesta acerca de las herramientas que se estudian. Ésta se efectúa sobre una muestra de 10 profesionales que trabajan en el ámbito del testing y la calidad de software.

Previamente a la realización de la encuesta se pide un par de datos acerca del encuestado y son:

- Formación académica
- Puesto profesional que desempeña

Estos datos se utilizaran para dar bagaje y credibilidad al cuestionario y a los resultados que arroje.

A continuación, se muestra la estructura que presenta el cuestionario que se entrega.

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tabla 66 - Valores encuesta

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación y puesto técnico que ocupa:

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación				
Portabilidad				
Estética				
Facilidad de uso				
Seguridad				
Mantenibilidad de los test				
Ayuda y soporte				
¿Cumple sus objetivos? (SI ó NO)				

Tabla 67 - Encuesta

En el Anexo B, se adjunta la transcripción de las valoraciones de todos cuestionarios realizados.

La media de satisfacción de las diferentes herramientas, para cada uno de los encuestados (E) es:

	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10
HP UFT	4,3	3,6	3,9	4,3	3,9	3,9	4,3	4	3,6	3,9
HP LoadRunner	4,3	3,4	4	4,3	3,9	3,7	3,7	3,7	3,4	3,3
Selenium IDE	2,9	2,3	__	3,9	2,3	2,4	3	3,7	2,6	3
Apache JMeter	1,7	2,7	2,6	3,4	2,6	3	1,7	3,4	3	1,7

Tabla 68 - Valor medio herramientas

Por lo tanto, partiendo de estos valores podemos sacar la valoración media general de la encuesta.

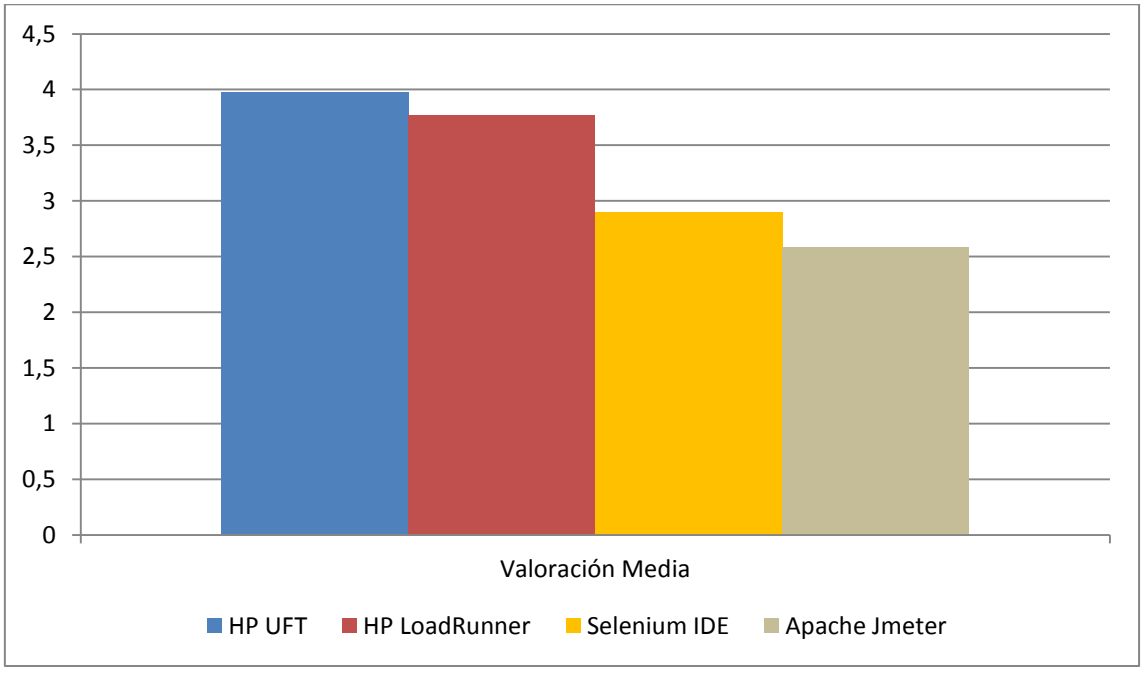


Figura 67 - Valoración media herramientas

El resultado no ofrece dudas, las herramientas de HP son las mejor valoradas y la por valorada es JMeter, posiblemente porque es la que requiere más formación y tiempo para aprender a sacarle el máximo rendimiento.

Ahora, se realiza el cálculo de la desviación típica por producto de los valores recogidos. Esto nos permitirá ver cómo están distribuidos los datos alrededor de la media.

Herramienta	Desviación típica
HP UFT	0,262
HP LoadRunner	0,356
Selenium IDE	0,583
Apache JMeter	0,669

Tabla 69 - Desviación típica

No existe demasiada variabilidad en ninguno de los casos y son bastante uniformes, sobre todo en el caso de las herramientas de pago. Por tanto podemos dar por buena la muestra ya que no se dispersa demasiado.

Ahora vamos a valorar la última pregunta que se realiza en el cuestionario y que es contestada con SI o NO. Gracias a estas respuestas, podemos analizar si las cualidades de las herramientas cumplen las necesidades del usuario.

	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10
HP UFT	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI
HP LoadRunner	SI	SI	SI	SI	SI	SI	SI	SI	SI	SI
Selenium IDE	NO	NO	__	SI	NO	NO	NO	NO	SI	SI
Apache JMeter	NO	SI	NO	SI	NO	SI	SI	NO	SI	NO

Tabla 70 – Valoración final herramientas

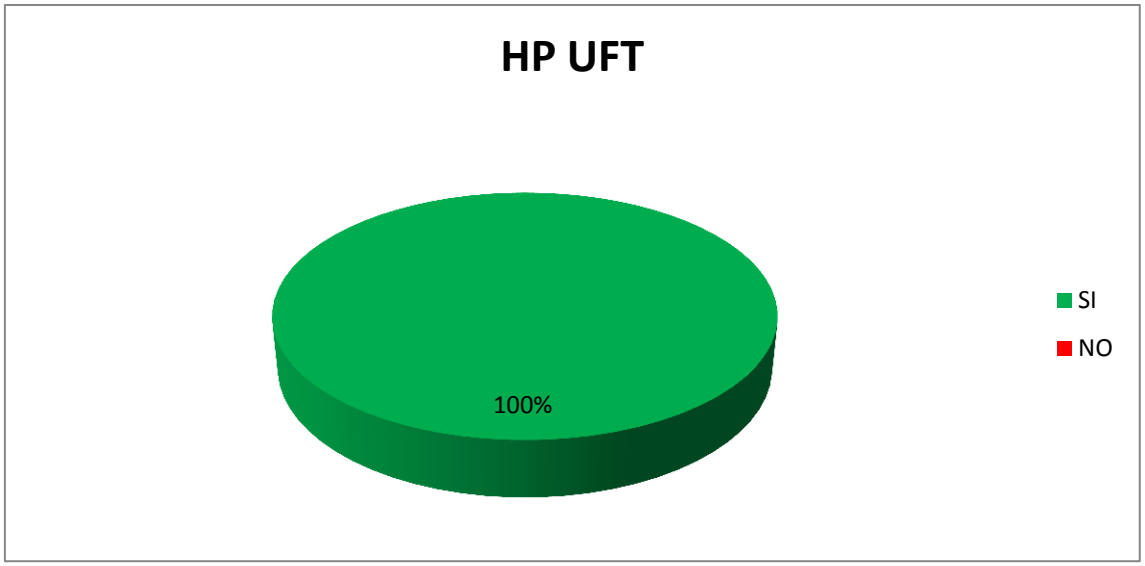


Figura 68 – Gráfico Objetivos HP UFT

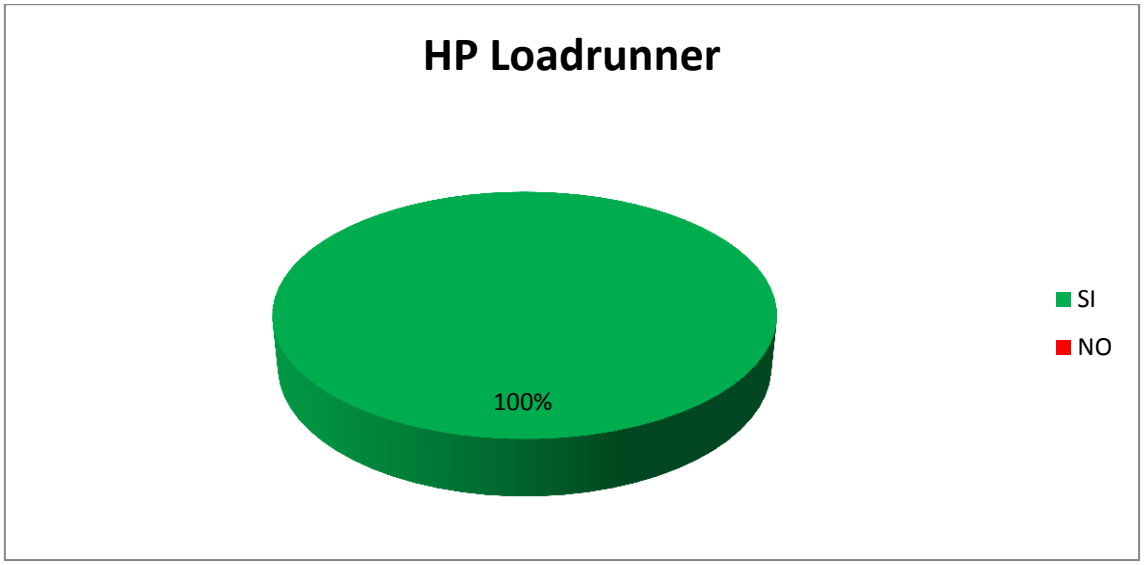


Figura 69 – Gráfico Objetivos HP Loadrunner

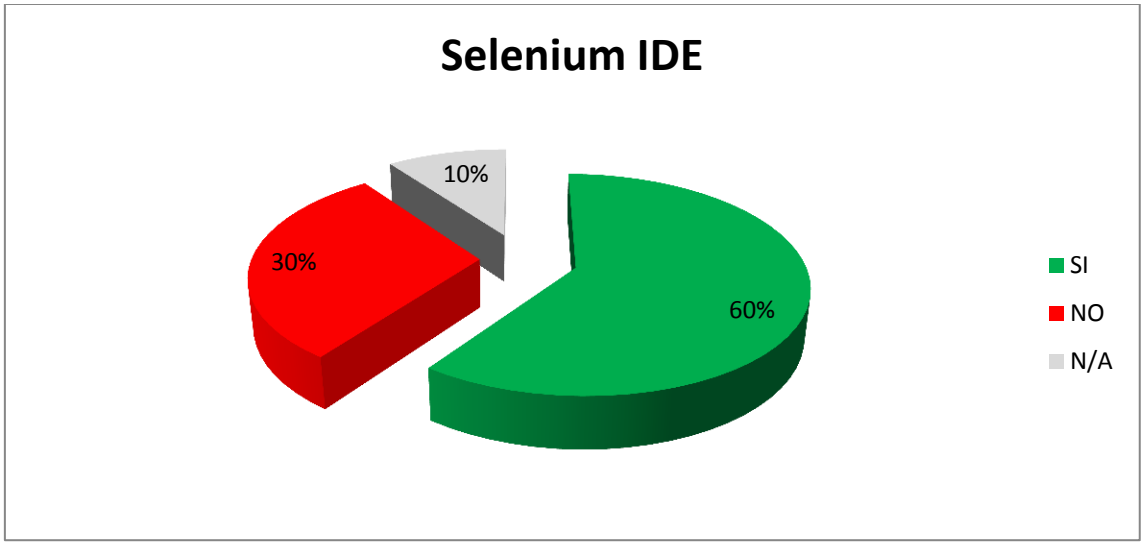


Figura 70 – Gráfico Objetivos Selenium IDE

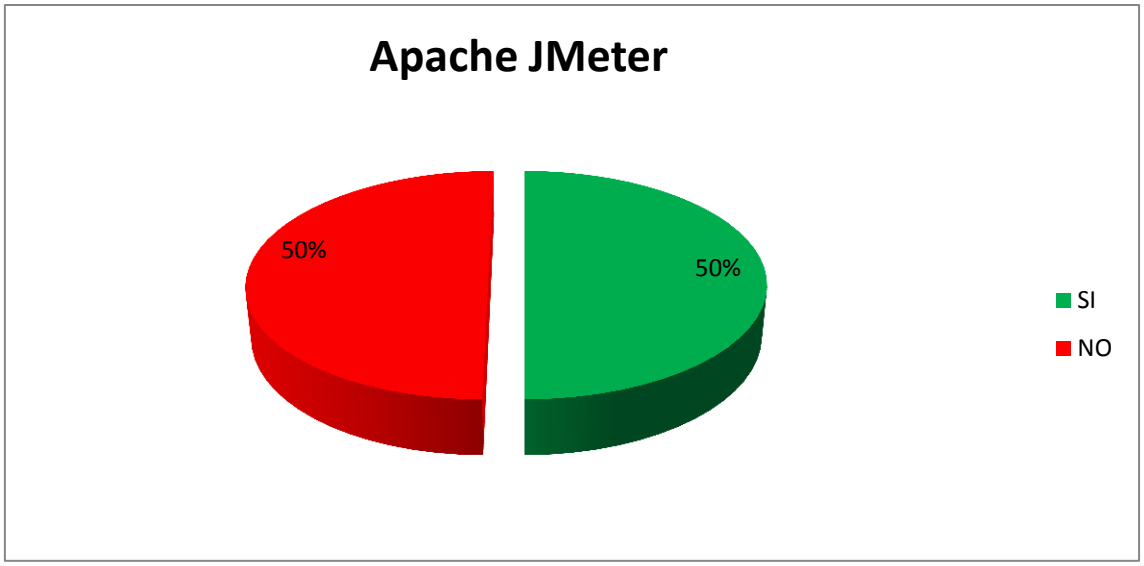


Figura 71 - Gráfico Objetivos JMeter

Las dos herramientas de pago satisfacen las necesidades del usuario de manera clara, ya que el 100 % de los encuestados ha valorado positivamente la cobertura ofrecida por estas suites al usuario. Este tipo de software al tener un mayor nivel de calidad, mejor interfaz de usuario y mayores capacidades, consigue ganarse al usuario que consume estos productos.

Sin embargo, en las herramientas gratuitas hay más diversidad cuando se valora el cumplimiento de los objetivos de cada persona. Ya que Selenium arroja un 60 % de contestaciones positivas, teniendo en cuenta que existe otro 10 % correspondiente a una de las personas que no la valoró por desconocerla. Por tanto, se puede valorar positivamente esta herramienta para el usuario. Y algo por debajo se encuentra JMeter con un 50 % de respuestas satisfactorias y valor igual de respuestas negativas.

Por supuesto, las herramientas gratuitas al disponer de una interfaz de usuario menos agradable, menos marketing, documentación, etc., implica que el usuario sea reticente ante este tipo de software. Sin embargo, el potencial de estas herramientas no tiene techo ya que se puede seguir desarrollando y mejorando continuamente. Por ello, en muchas ocasiones si se tuviese un conocimiento previo más amplio acerca de estos productos se verían con mejor perspectiva.

Además de los aspectos ya estudiados hay que añadir algunas de las diferencias técnicas. A continuación se destaca, a modo resumen las principales diferencias entre HP UFT y Selenium.

	UFT	Selenium
Lenguajes soportados	VB Script	Java, C#, Ruby, Python, Perl PHP , JavaScript
Navegadores soportados	Google Chrome, Internet Explorer , Firefox, Safari	Google Chrome , Internet Explorer, Firefox , Opera , HtmlUnit, Safari
Soporta Aplicaciones basadas en ventanas que no sean web	Si, tiene add-ins para Emuladores, citrix, Java, Siebel etc. Además de la grabación Insight Object.	No
Entornos Soportados	Windows	Windows , Linux , Solaris OS X , Otros (como Brower & JVM o JavaScript)
Soporta sistemas operativos para dispositivos móviles	En otro software de la misma suite	Android , iPhone & iPad, Blackberry , Headless WebKit
Framework	HP Quality Center y Performance Center integrados dentro de HP ALM	Selenium + Eclipse + Maven, ANT+ Jenkins, Hudson & its plug-ins, Cruise Control + TestNG + SVN Nunit, Junit, TestNG, Behat + Mink , unitest, pyunit, py.test, robot framework, Rspec, Test::Unit , Cinnamon5
Reconocimiento de objetos	Incorpora un repositorio de objetos	IU Mapas y estrategias de localización de objetos como XPath Element ID
Alto nivel técnico/programación	No es necesario	Necesario
Depuración de errores	Alta. Breackpoints, watches, ejecución modo mantenimiento	Baja. Breackpoints.
Resultado e Informes de análisis	Results Viewer, Screen Recorder y soporte en QC.	Integrado con Jenkins si los soporta
Recursos Hardware consumidos durante la ejecución de los test	Muchos, ya que UFT por sí solicita muchos requerimientos.	Muchos, debido a que su utilización está ligada a Mozilla Firefox
Velocidad de ejecución	Normal	Rápido
Ayuda y Documentación	Mucha dentro de la herramienta y online.	Poca y online
Parametrizaciones	Sí	No
Soporte de usuario	Sí	No

Tabla 71 - Comparativa cualitativa UFT y Selenium

Ahora se hace una comparativa similar pero entre las herramientas de carga/rendimiento.

	LoadRunner	Apache JMeter
Protocolos soportados	.NET, Ajax - Click and Script, C VUsers, Citrix ICA, COM/DCOM, DNS, EJB, Flex, FTP, LDAP, IMAP, Java over HTTP, Java Record Replay, Java Vuser, MAPI, MMS, Mobile Application - HTTP/HTML, OBDC, Oracle - 2Tier, Oracle - Web App 11i, Oracle NCA, POP3, RDP, RTE, SAP, SAP GUI, Siebel Web, SAP, SAP GUI, Siebel Web, Silverlight, Silverlight, SMP, SMTP, TruClient Mobile, TruClient Native Mobile, Web, Web HTTP/HTML, Web HTTP/HTML, Web Service, Windows Sockets TruClient.	HTTP,HTTPS, FTP, SOAP/XML- RPC, JDBC, LDAP, JMS, Mail-POP3(S) y IMAP(S)
Informes de análisis	Si, mucha diversidad de gráficos e informes.	Sí, menos variedad en gráficos y resúmenes.
Depuración	Sí, vista extendida del log, debug en el generador de scripts.	Vista extendida del log
Simulación de velocidad de conexión de usuarios	Sí	No, pero puede programarse con JAVA
Parametrizaciones	Sí	A través de Interfaz
Monitorización y análisis en tiempo real	Sí	Sí, se amplía cuando coopera con otros módulos
Generación ilimitada de carga	No	Sí
Soporta gran rendimiento de descarga	Sí	No
Alto nivel técnico/programación	Necesario	Necesario
Multiplataforma	No	Sí
Transacciones definidas por el usuario	Sí	A través de plug-ins
Soportar la velocidad de conexión variable del ancho de banda	Sí	No

Tabla 72 - Comparativa cualitativa LoadRunner y JMeter

5.2.2 Análisis cuantitativo.

Inicialmente, la comparativa cuantitativa entre herramientas se centra en el coste de adquirirlas. Este aspecto es de gran importancia ya que las comerciales requieren el pago de licencias para poder ser utilizadas más allá de la versión de prueba. Sin embargo, las libres que empleamos no necesitan pagos para poder utilizarse y modificarse, en ningún momento.

La versión de prueba de HP UFT permite utilizarla durante 30 días de manera gratuita. Mientras que la de HP LoadRunner tiene una limitación de 50 VUsers. Si queremos ampliar estas restricciones se debe de adquirir las licencias. El precio aproximado (en euros), el tiempo de duración de dicha licencia y los permisos que se otorgan por la compra de las mismas, se muestran a continuación.

	Precio (€)	Permisos	Duración
HP LoadRunner	1.000	500 Vusers	1 día
	1.400	1000 Vusers	1 día
HP UFT	24.000	10 usuarios	3 meses
	37.000	10 usuarios	6 meses
	49.000	10 usuarios	9 meses

Tabla 73 - Comparativa precios y permisos

El valor de los VUsers se refiere a los usuarios virtuales que se utilizan y ejecutan concurrentemente en el escenario de pruebas (Controller). Sin embargo, el dato de los usuarios de UFT se refiere a usuario reales que pueden utilizar la herramienta de manera concurrente.

Se observa que para los productos privativos el coste de las licencias es bastante alto. Por ello, dependiendo del volumen de pruebas, de la infraestructura y del poder adquisitivo del comprador, se seleccionara software libre o de pago.

Los requisitos de sistema necesarios para las suites de HP, tanto en la instalación como en la implementación y ejecución (memoria, CPU, etc.), es bastante alta, sobre todo en HP LoadRunner. Esto se debe a que estas herramientas tienen muchos módulos y componentes.

En contraposición, las herramientas gratuitas necesitan menos recursos en la instalación y durante su utilización nunca superan las necesidades del software comercial. Estos puntos ya han sido analizados en capítulos anteriores.

Una muestra significativa de la evolución y de los módulos y capacidades que albergan las herramientas es la diferencia de versiones y de tamaño en disco duro que ocupa cada paquete software.

Herramienta	Versión	Tamaño en disco
HP UFT	12.51	1,19 GB
HP LoadRunner	12.50	1,78 GB
Selenium IDE	2.9.0	752 KB
Apache Jmeter	2.13	82,2 MB

Tabla 74 - Versiones y tamaños disco duro

En la tabla anterior sólo se indica el tamaño del software en sí, a parte alguno de ellos necesita la instalación de ciertos complementos para funcionar correctamente.

Por otra parte los tiempos de ejecución de las pruebas, son más bajos en los útiles libres. En gran medida debido a que el procesamiento de datos, los chequeos y al generación de resultados e informes es más pesada en el software de HP. Este estudio se realizó en el capítulo 5.

El número de usuarios virtuales que se pueden ejecutar correlativamente en HP Loadrunner también tiene limitaciones, sin embargo en JMeter no tiene limitaciones preestablecidas, sino que vendrán definidas por la plataforma e infraestructura del software que se prueba.

	Mínimo VUsers	Máximo VUsers
HP LoadRunner	0-999	1000 VUsers
Jmeter	0	Sin límite

Tabla 75 - Comparativa carga VUsers

5.2.3 Resumen de los resultados obtenidos.

Tras el análisis realizado sobre las herramientas, las pruebas abordadas y los resultados obtenidos, se puede realizar un resumen con los puntos más destacables.

El primer punto y que provoca una mayor diferenciación, es el tema de la obtención de las herramientas tanto en relación al tiempo como al dinero que conlleva. Tanto Jmeter como Selenium IDE se tratan de herramientas libres y gratuitas, es decir son de código abierto y se pueden adquirir fácilmente, de manera rápida y sin la necesidad de realizar un pago de una licencia. En contraposición, tanto HP UFT como HP LoadRunner necesitan tiempo para ser adquiridas (rellenar formularios, registrarse en HP, recibir emails de confirmación, etc.), y unido con el alto coste de licencias de uso, determina en muchas ocasiones el utilizar una u otra herramienta. Todas estas herramientas se pueden adquirir desde sus respectivas páginas oficiales (ver referencias [18] a [21] del apartado Bibliografía).

En relación a los requisitos necesarios por cada herramienta, se puede concluir que las herramientas comerciales son las que más requisitos previos requieren. Esto se produce debido a que tienen más módulos y capacidades, y a que éstos son más pesados. Selenium IDE, es el software que menos recursos requiere ya que se trata de un complemento de Firefox y es el más “simple” tanto en apariencia, como potencial y patrones de análisis.

En cambio, en el terreno de los recursos que se consumen durante la grabación y/o ejecución, las relaciones son diferentes que en los requisitos previos. Y es que, durante la grabación en los diferentes productos software estudiados se consume un peso similar de procesos. Si el navegador web utilizado es Mozilla Firefox, éste es el que más recursos se lleva consigo, más incluso que el propio software de automatización.

Sin embargo, cuando se realiza la ejecución este consumo se dispara, especialmente para las aplicaciones de pruebas de rendimiento. Esto dependerá del número de navegaciones paralelas que se realicen y del número de inyectores de carga, pues habrá tantos navegadores corriendo como navegaciones ejecutándose a la vez. Para solucionar esta carga de recursos (CPU, memoria etc.) se requiere una infraestructura

adecuada al volumen de la prueba que se quiera realizar. HP Loadrunner es el que más necesidades y capacidad del sistema requiere, para poder ejecutar sin saturar las máquinas que intervienen en una prueba.

Dentro del ámbito de la automatización de pruebas funcionales, se observa que HP UFT sólo tiene VBScript como lenguaje de desarrollo mientras que Selenium tiene Java y más variedad al respecto. Esto es bueno para el tester, ya que no se le limita a la utilización de un lenguaje para poder utilizar y desarrollar scripts en la herramienta.

Pero no todo es positivo para Selenium IDE, y es que la mayor restricción de la que dispone es que solo soporta aplicaciones Web. Mientras que HP UFT tiene un mayor potencial en este aspecto, ya que es capaz de grabar y ejecutar sobre un abanico mucho más amplio de plataformas y protocolos.

En el dominio de las pruebas de carga hay que destacar que Loadrunner es la más completa. Esto se debe a que es la que más protocolos soporta y que más opciones aporta a la hora del análisis de resultados y de errores encontrados (logs), pero tiene un límite de carga (999 Vusers) vinculado al pago de licencias por demanda. Sin embargo, Jmeter además de poder probar sobre varios tipos de servidores también aporta un nivel adecuado, aunque no tan exhausto, de análisis de resultados y errores. Además, no tiene límite de carga, es capaz de ejecutarse en modo batch y multiplataforma y consumen menos recursos de Loadrunner.

A partir de todos estos aspectos recogidos y de otras características estudiadas, se pueden recopilar los puntos fuertes y las desventajas de utilizar cada una de las herramientas.

- **HP UFT**

  Soporta gran variedad de tecnologías. Se basa en el reconocimiento de objetos y tiene un repositorio de objetos que permite una modificación y actuación de los mismos de una manera más simple. Posee una interfaz cercana y fácil de utilizar. No requiere mucho nivel de conocimientos de programación. El sistema de análisis de resultados, grabación de evidencias de las ejecuciones

y depuración, es extenso y eficiente. Tiene un servicio de soporte disponible y un módulo de ayuda muy completo.

Requiere el pago de licencia a partir del primer mes de muestra gratuita. No permite la integración con procesos de desarrollo, tan sólo con los requeridos por HP. No permite su utilización fuera del sistema operativo Windows. Como lenguaje de programación sólo soporta VBScript.

- **Selenium IDE**

La principal ventaja es que es de código abierto y permite utilizar lenguajes orientados a objetos, como Java entre otros. Se puede adquirir gratuitamente y es posible su integración con herramientas externas y diferentes librerías. También es soportado en diferentes sistemas operativos.

Como puntos débiles, sobresale la capacidad de trabajo reducida a entornos web. No permitiendo trabajar con cualquier otra plataformas o tecnología.

Además, esta herramienta no permite el almacenaje y administración de objetos. Requiere de un nivel alto de programación por parte del usuario, haciendo más complicado la creación de scripts. Menor capacidad de realizar comprobaciones y checkpoints. El soporte técnico es on-line, de tipo comunidad, y no podemos disponer de ella si conexión.

- **HP LoadRunner**

Es compatible y funciona con la mayoría de protocolos. No necesita en el servidor bajo prueba ya que utiliza monitores nativos. Tiene una excelente interfaz de control, monitorización y análisis, donde se pueden ver informes con tablas y gráficos coloreados fáciles de entender. Permite definir fácilmente correlaciones, transacciones y grabar en modo click and script. Tiene magníficos tutoriales y soporte de la herramienta. Permite la ejecución en modo normal, en modo mantenimiento y en modo actualización.

Como desventajas se resalta la obligación de pagar licencia para utilizar más de 50 Vuser. Tiene necesidad de un completo paquete de recursos de sistema. Es extensible sólo con software de la misma suite, como ALM-Performance Center,… La grabación se traduce en líneas de código que requieren de un nivel medio-alto de programación por parte del usuario.

- **Jmeter**

Admite la posibilidad de realizar una portabilidad completa, permite el muestreo concurrente por muchos hilos y el muestreo simultáneo de diferentes funciones por grupos de hilos separados. También tiene la opción, al hacer almacenamiento en caché, de realizar un análisis de datos sin conexión. Es altamente extensible. No tiene restricción preestablecida respecto al volumen de carga preestablecido.

Como claras insuficiencias está que al ser open Source no tiene garantía de soporte o de mayor desarrollo. Además necesita de un nivel de conocimientos altos por parte del usuario en relación a la configuración del cliente-servidor, del lenguaje de programación, del análisis de resultados etc. Ya que estos son algo complejos y si no se miden adecuadamente se pueden sacar conclusiones erróneas (capacidades, cuellos de botellas etc.)

Basándome en todo lo los expuesto anteriormente y en mi experiencia profesional acerca de los diferentes productos software, utilizaría las siguientes herramientas según el objeto de la prueba.

Para automatizar pruebas funcionales, si lo que se desea probar es una página web se pueden utilizar cualquiera de las dos (HP UFT o Selenium). Sin embargo, si en estas navegaciones se desean capturar objetos no basados 100% en web utilizaremos HP UFT. Ya que, por ejemplo en las pruebas realizadas, se ha observado que si pasamos por imágenes, mapas, etc. Que aparecen sobre una ventana del navegador pero que no son objetos 100% web, entonces Selenium no lo captura. Esto también ocurre cuando pasamos por funcionalidades que se apoyan en otras arquitecturas, como por ejemplo el caso que se ha hecho del envío de correo al PIC, en el cual sólo se ha podido grabar con HP UFT.

Si tenemos una página web donde se puede grabar completamente un caso de prueba en ambas herramientas, se seleccionará Selenium IDE cuando deseamos realizar pruebas más simples, en aplicaciones muy estables y de capacidades aceptables. Sin embargo, cuando se quiera realizar pruebas regresión de un gran volumen de casos, con dependencias entre casos, reutilización de funcionalidades etc. Se utilizará HP UFT.

Obviamente, si además se quiere automatizar aplicaciones GUI que no está basado en web, como por ejemplo Terminales Virtuales, Java, Visual Basic, .NET, Citrix etc., se utilizará HP UFT. Además esta herramienta también es capaz de trabajar con el modo API Test (sin interfaz de usuario) y automatizar por ejemplo web services.

Con respecto, a las pruebas de carga/rendimiento. Si se ejecutan pruebas puntuales y que no van a perdurar mucho en el tiempo se utilizará Jmeter, ya que para este tipo de pruebas funcionan perfectamente para este tipo de tests y no requieren del pago de licencias. Mientras que si se aborda un volumen amplio de pruebas y que se apoyen en una arquitectura compleja, se utilizará HP Loadrunner. Gracias a esta herramienta se puede implantar un sistema de pruebas apropiado y se pueden visualizar informes de resultados muy completos y exportables, gracias al módulo de HP Analysis.

Capítulo 6

Planificación y presupuesto.

En este capítulo se detalla a planificación llevada a cabo para la elaboración de cada una de las etapas y tareas abordadas hasta finalizar este proyecto. Todo ello se muestra mediante un diagrama de Gantt para que sea más visual.

Seguidamente, también se desglosa el presupuesto necesario para elaborar el estudio, con los costes derivados de realizar toda la casuística del mismo.

6.1 Planificación.

A continuación se muestra la planificación, es decir un resumen de las fases por las que se ha pasado para realizar el proyecto y el tiempo invertido en cada una de ellas.

Se ha excluido el periodo de tiempo de las vacaciones, correspondiente a la tercera y cuarta semana de Julio.

Fase	Fecha Inicio	Fecha Fin	Total días
Propuesta	08/04/2015	08/04/2015	1
Estudio Viabilidad	09/04/2015	15/04/2015	7
Documentación Inicial	16/04/2015	14/05/2015	29
Análisis de herramientas	15/05/2015	03/06/2015	20
Estudio de las herramientas seleccionadas	04/06/2015	10/07/2015	37
Implementación y ejecución pruebas	27/07/2015	18/09/2015	54
Análisis resultados	19/09/2015	21/09/2015	3
Documentación	22/09/2015	13/10/2015	22

Tabla 76 - Planificación de fases

6.1.1 Diagrama de Gantt.

Se representa gráficamente la anterior planificación de tareas a través de un diagrama de Gantt, para que se pueda relacionar de manera visual los trabajos realizados y los tiempos transcurridos en cada una de las etapas.

Las etapas donde más tiempo se ha invertido son:

- Estudio de las herramientas que se han seleccionado, que incluye desde la adquisición de las herramientas hasta el aprendizaje de utilización y desarrollo de las mismas.

- Implementación y ejecución de las pruebas, la cual comienza desde el planteamiento de qué pruebas cuantitativas y cualitativas queremos realizar, plataformas a probar, diseño e implementación de las mismas, ejecución, realización de encuestas, etc.

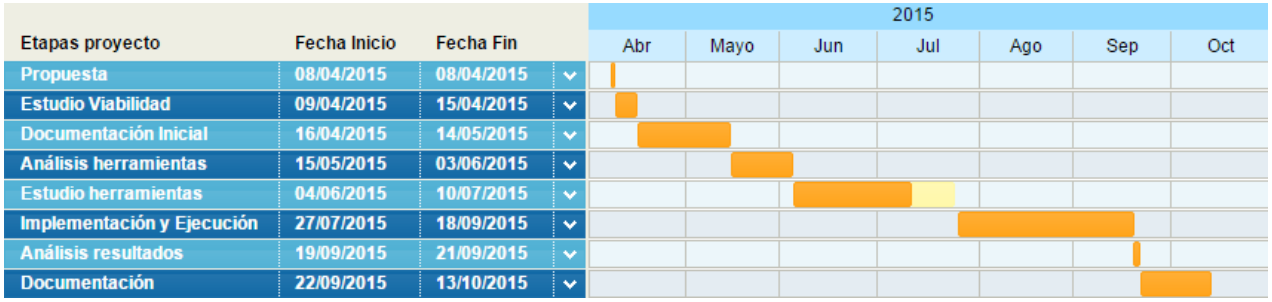


Figura 72 - Diagrama Gantt PFC

6.2 Presupuesto.

El cálculo del coste del trabajo realizado se dividirá en relación a los tiempos y el material utilizado para la elaboración del mismo.

Las tarifas aparecerán sin IVA a excepción del resumen de costes donde aparecen los precios totales del estudio, con el IVA incluido.

6.2.1 Resumen del tiempo dedicado.

Gracias a la tabla planificación de tareas y el diagrama de Gantt del anterior punto, podemos extraer de una manera sencilla y exacta los datos relativos a los tiempos invertidos a cada una de las etapas.

Se utilizará como regla de equivalencia una media de 5 horas por día trabajado.

Fase	Total días	Total horas
Propuesta	1	5
Estudio Viabilidad	7	35
Documentación Inicial	29	145
Análisis de herramientas	20	100
Estudio de las herramientas seleccionadas	37	185
Implementación y ejecución pruebas	54	270
Análisis resultados	3	15
Documentación	22	110

Tabla 77 - Tiempo dedicado

El total de días dedicados al proyecto es de 173 jornadas de trabajo que se traducirían en un total de 865 horas invertidas.

6.2.2 Desglose: Coste de personal.

La duración del proyecto ha sido alrededor de unos 6 meses. Durante este tiempo se ha realizado un trabajo total de unas 865 horas, divididos del siguiente modo según la especialidad de la persona que realiza el trabajo.

El rol de diseñador será el encargado de diseñar los casos de prueba que se van a realizar y documentarlos, para que posteriormente el ingeniero de pruebas software los implemente con el software de automatización y ejecute las pruebas para su siguiente análisis y reporte de todo lo observado.

Por lo tanto se estipula, conforme al rol desempeñado y el mercado actual con jornadas laborales de 8 horas, un precio mensual para el desarrollador de casi unos 1800 € (precio por hora alrededor de 10 €) y un sueldo mensual de unos 2100 euros para un ingeniero de pruebas (precio por hora alrededor de 12 €).

Se le aplica al diseñador un total de unos 15 días y al ingeniero de pruebas el resto, 158.

Apellidos y Nombre	Categoría	Dedicación (hombre/mes)	Coste hombre/mes (€)	Coste (€)
Álvaro Pérez, Daniel	Diseñador	1	1800 €	750
Álvaro Pérez, Daniel	Ingeniero de pruebas	1	2100 €	9.480
			TOTAL:	**10.230**

Tabla 78 - Coste Personal

6.2.3 Desglose: Coste de Hardware.

Para la elaboración del estudio ha sido necesario utilizar un terminal portátil. Éste se ha empleado tanto para la instalación del software que se ha utilizado y realizar las pruebas pertinentes, como para realizar la documentación, gráficos, etc.

El coste imputable del mismo se hace mediante la siguiente fórmula de cálculo de la amortización:

$$\frac{\mathrm{A}}{\mathrm{B}} xCxD$$

A = nº de meses desde la fecha de facturación en que el equipo es utilizado

B = periodo de depreciación (60 meses)

C = coste del equipo (sin IVA)

D = % del uso que se dedica al proyecto (habitualmente 100%)

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Periodo de depreciación	Coste imputable (€)
Portátil DELL Intel ® Core ™ i3-2310 M CPU 2.10 GHz 4 GB RAM	860	100	6	60	86
				TOTAL:	**86**

Tabla 79 - Coste Hardware

6.2.4 Desglose: Coste de Software.

A continuación se hace una declaración del coste del software comercial utilizado y el respectivo precio por el uso de las licencias necesarias. El software libre y gratuito no se incluye en este resumen de gastos.

Descripción	Coste (€)
Windows 7 Professional	235
Microsoft Office Professional Plus 2010	399
HP Unified Functional Testing	808
TOTAL:	**1.442**

Tabla 80 - Coste Software

El coste de utilizar HP Unified Functional Testing se desprende de utilizar el UFT durante un mes más, a parte de los 30 días que HP da gratuitamente al descargarlo por primera vez. Y para ello he necesitado hacerme con una licencia de un usuario por un mes de duración (para calcular este coste se ha hecho una extrapolación del precio de la licencia para 10 usuarios concurrentes por mes, a un sólo usuario)

6.2.5 Resumen de costes.

El coste total que se deriva de todo el trabajo realizado se desglosa en el siguiente cuadro. Se añade un 20% de costes indirectos para cubrir los riesgos del proyecto y los gastos que no han sido tenidos en cuenta al realizar el presupuesto.

Se detalla el coste total con y sin I.V.A.

Descripción de Costes	Coste Total (€)
Personal	10.230
Hardware	86
Software	1.442
Costes indirectos (20%)	2.351,6
Total sin I.V.A	14.109,6
Total (21% IVA incluido)	**17.072,6**

Tabla 81 - Resumen de costes

El presupuesto total de este proyecto sin IVA asciende a la cantidad de **14.109,6 euros** y aplicando el 21% de I.V.A asciende a la cantidad de **17.072,6 euros**.

Capítulo 7

Conclusiones y Líneas

En este último apartado del proyecto es necesario realizar una visión global de todo lo que se ha estudiado, de todo lo que se ha aprendido y de los objetivos que se han alcanzado. Tanto a nivel del estudio realizado como a nivel personal.

Posteriormente se expondrán los proyectos de futuro y las líneas a realizar como consecuencia del trabajo elaborado.

7.1 Conclusiones.

7.1.1 Conclusiones del Proyecto.

El objetivo principal del proyecto, fundamentado en la realización de una comparativa entre las herramientas, comerciales y gratuitas, de automatización más potentes y usadas en el mercado actual, se ha alcanzado satisfactoriamente. Gracias a este estudio, se han aportado tanto argumentos a favor como en contra para cada una de ellas.

Según los resultados realizados, y basándonos en el potencial y prestaciones que ofrecen estas herramientas al usuario, se llega a la conclusión de que los productos de pago son los más robustos y mejor valorados.

Esta valoración, realizada también por profesionales que utilizan herramientas de este bagaje día a día, se debe a que este tipo de software cuida de manera notoria aspectos muy importantes para el consumidor como la imagen, su dinamismo, las ayudas y soportes online etc. Todo esto, unido a que son las más completas a nivel de protocolos, plataformas y arquitecturas a probar, las hacen las reinas del testing.

Sin embargo, una carga muy pesada que tienen estas herramientas es el coste de las licencias para su uso. Este hándicap es el que, en muchas ocasiones equilibra la balanza e incluso la puede llegar a inclinar a favor de las libres y/o gratuitas.

Las herramientas de libre adquisición, una vez estudiadas, probadas y analizadas, aportan muchas más funcionalidades de las que a priori presumían. Esto se debe a que este tipo de software necesita de un nivel técnico y de programación mayor que en las herramientas comerciales. Teniendo este conocimiento es posible realizar pruebas más complejas y robustas.

También esta clase de software permite, por parte del desarrollador, una mayor flexibilidad y adaptabilidad tanto en su configuración, como en sus funcionalidades. Posibilitando que se acople mejor a sus necesidades.

En contraposición, cabe resaltar que el software no privativo es más inestable ya que el soporte sobre el que se establecen, en ocasiones no es el más idóneo.

Además del estudio comparativo se han alcanzado otros propósitos derivados del tema principal:

- ✓ Entender la importancia de la calidad, del testing y de las herramientas de automatización dentro del ciclo de vida software.

- ✓ Realizar un estudio de mercado acerca de las herramientas existentes y de las más potentes y utilizadas.

- ✓ Acercar este tipo de herramientas al lector, llegando incluso a marcar pequeñas pautas a la hora de la elaboración de pruebas.

Estas cuestiones, a pesar de tratarse de temas secundarios, aportan un extra de información que ayuda a conocer mejor el ámbito y los objetivos de las herramientas de automatización de pruebas.

7.1.2 Conclusiones Personales.

Debido a que cada día se reclama una mayor calidad en los productos software, este proyecto surgió como una necesidad al respecto. Para conseguir ese nivel de calidad, se debe disponer de un buen grupo de herramientas en las que apoyarse. Y aquí surge la pregunta, ¿qué herramientas son las idóneas?

Pues bien, partiendo de la base de que el software perfecto no existe y tras analizar los resultados obtenidos, concluyo que la elección de una herramienta u otra dependerá de quien vaya a utilizarla y en que ámbito. Si van a realizarse pruebas por parte de una sociedad con buen nivel adquisitivo y éstas van enfocadas hacia un gran volumen de tests, conviene utilizar software comercial. Sin embargo, si no se dispone de suficientes medios económicos o si se realizan sólo pruebas puntuales, conviene invertir en formación y testear con productos libres.

Además, gracias a todo lo estudiado, asimilado y probado durante este proyecto, he aprendido a dar más importancia y confianza a las herramientas libres y/o gratuitas, y he observado cómo con paciencia, tiempo y trabajo se puede sacar más provecho a este tipo de software.

7.2 Líneas Futuras.

Dentro del ámbito laboral he realizado una propuesta para introducir software no privativo orientado a realizar algunos tipos específicos de pruebas. De esta forma se liberarían licencias que tenemos de herramientas comerciales, ya que son necesarias y en ocasiones insuficientes para todo el grupo de testers.

También sería interesante ampliar este estudio dándole otros objetivos, tanto dentro del ciclo de vida de software (pruebas de código, de seguridad etc.) como hacia nuevas plataformas en auge (pruebas sobre terminales móviles, tablets etc.).

Anexo A

Curso rápido aprendizaje HP UFT + HP ALM.

Este anexo acerca de una manera más detallada y familiar el uso de la herramienta HP UNIFIED FUNCTIONAL TESTING y de HP APPLICATION LIFECYCLE MANAGEMENT, además de la integración entre ambos productos.

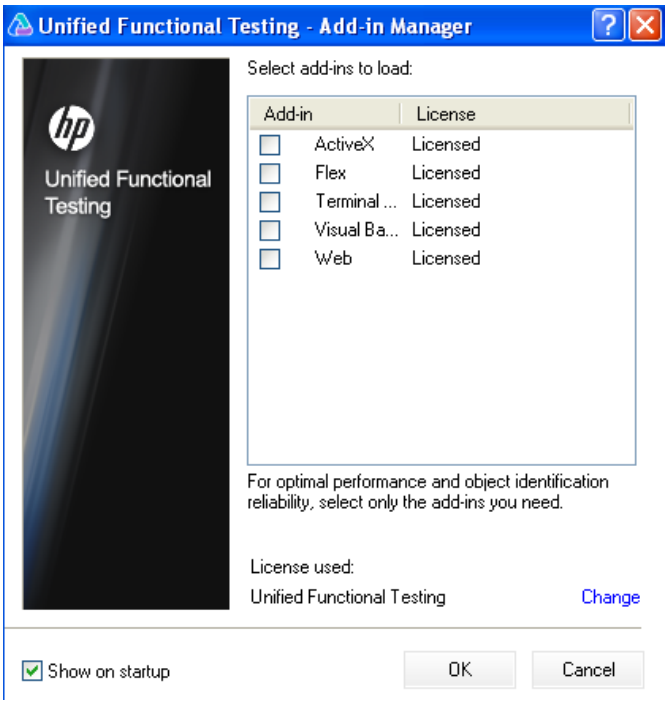
Se realiza un tutorial con una descripción de las principales funcionalidades, características y componentes de HP UFT y HP ALM, acompañados de capturas de pantalla de navegaciones por la propia interfaz del usuario.

Índice de contenidos

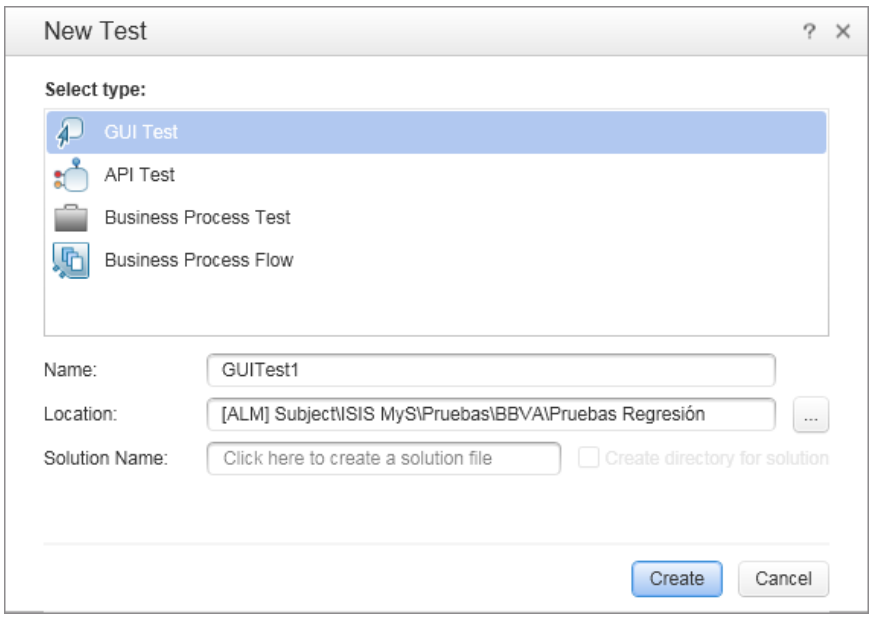
➢ Funcionalidades de UFT

Creación de scripts de pruebas

Lo primero que se debe hacer es abrir la herramienta HP Unified Functional Testing (UFT). Tras ejecutarlo, aparecerá una ventana como la siguiente, en la que se tiene que seleccionar los Add-ins que se quiere utilizar.



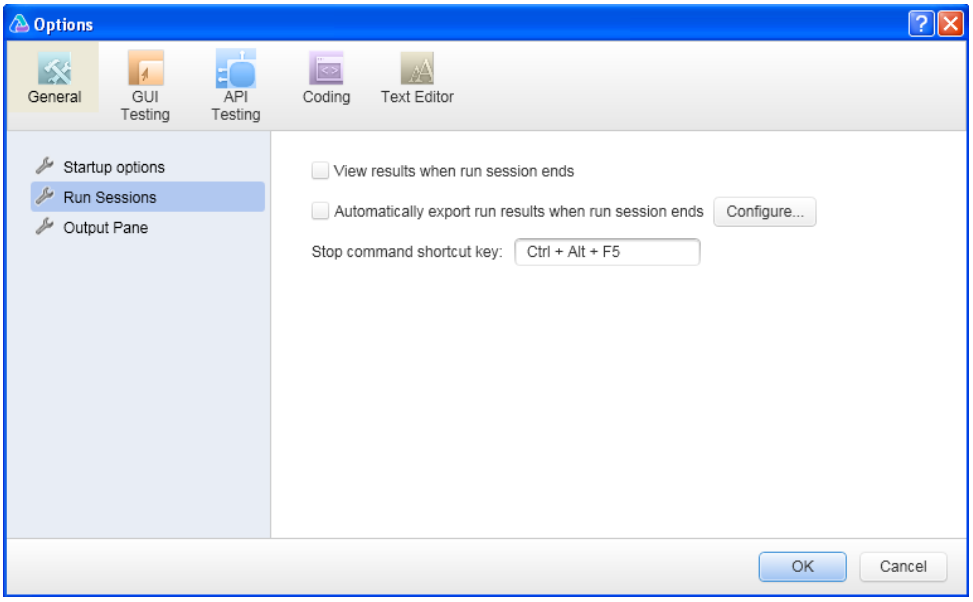
En UFT se accede a File > New > Test… En la ventana que aparece se selecciona el tipo de test que a crear (GUI Test, API Test…) y se pulsa sobre "Create".



Configuración de scripts de pruebas

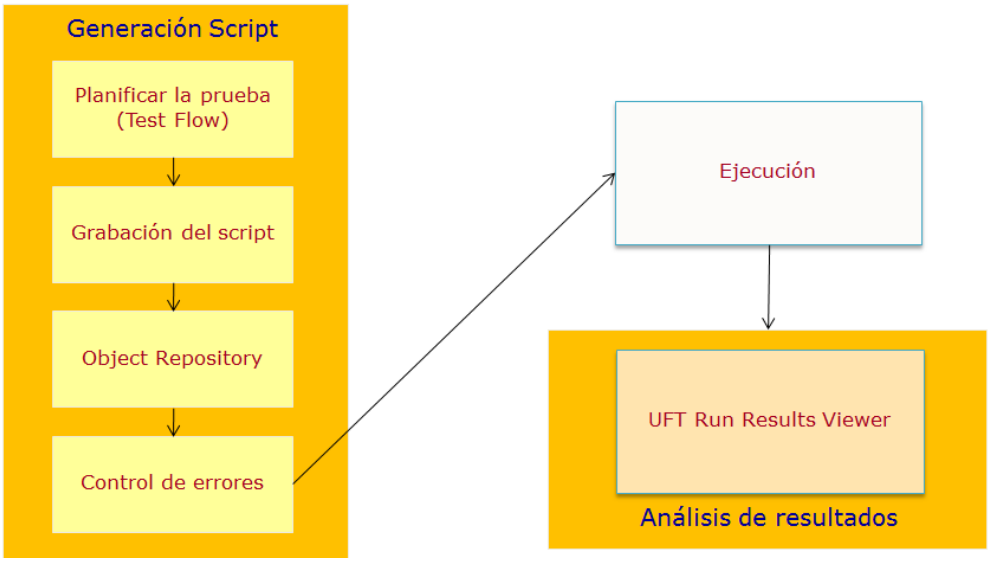
Desde el menú Tools > Options se accede a la ventana de opciones de configuración de UFT.

En ella se podrá configurar tanto opciones generales, como las propias de los diferentes tipos de test (GUI Testing, API Testing …) y de los add-ins que se hayan asociado a nuestro nuevo script.



Organización de scripts de pruebas

UFT permite la generación de un script automatizado, su ejecución y el análisis de los resultados obtenidos.



Estructura de scripts de pruebas

La estructura inicial de un script de prueba está conformada por un sólo Action.

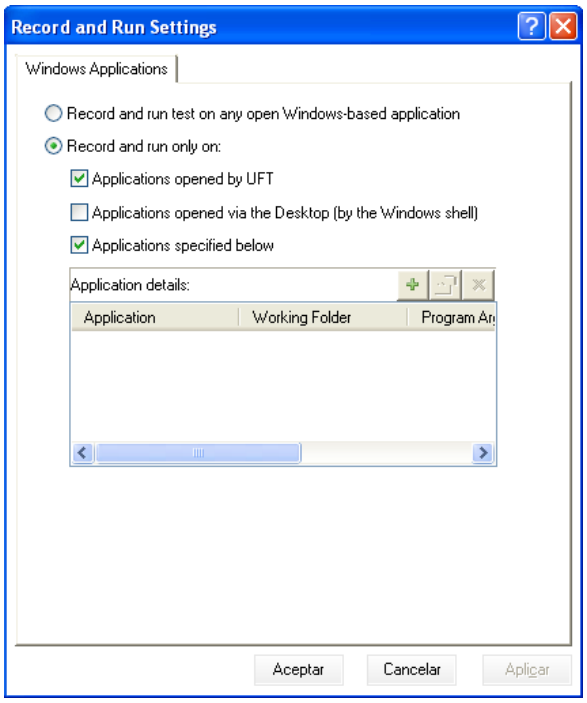
Desde el Test Flow se puede ver la estructura del script de una manera más visual.

Cuando se conforma un script automatizado, éste puede estar formado por varias acciones, las cuales se llamarán en el orden de ejecución deseado.

Esta modularización en Actions se realiza para poder llevar a cabo una encapsulación y reutilización más efectiva y ágil.

Modos de grabación

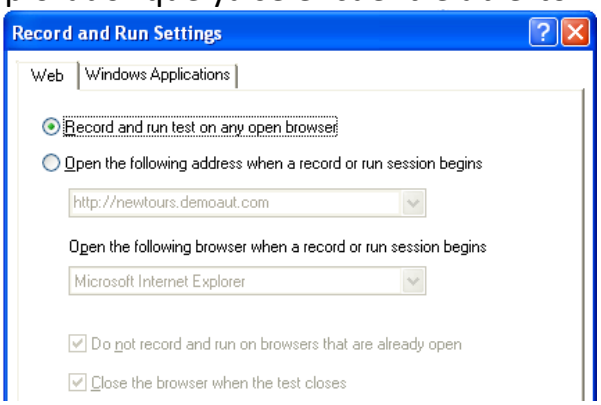
En UFT se accede a Record > Record And Run Settings… En esta ventana se da la opción de grabar o bien desde cualquier ventana que se encuentre abierta en Windows o bien desde la aplicación que se abre a través de UFT o desde una aplicación que se haya añadido previamente en la lista que aparece etc.



Se puede realizar la grabación de diferentes aplicaciones en diferentes plataformas. Si se realiza la grabación de alguna plataforma específica que requiera de un Add-in de UFT, se debe seleccionar antes de crear el nuevo Test.

Según los Add-in que se han seleccionado aparecerán otras opciones de grabación, aparte de las generales que ya se han visto, específicas para ellos.

- ✓ Por ejemplo, para Web en el apartado de Record and Run Settings aparecerá otra pestaña llamada Web en la que se podrá indicar si se desea arrancar la grabación abriendo automáticamente UFT un explorador con la url que se le indique o por el contrario que se realice la grabación a través de un explorador que ya se encuentre abierto.



La grabación del script se puede realizar abordando el ciclo de dos modos distintos, pero con un resultado similar:

- ✓ Pulsar el botón de grabar que se encuentra en Record > Record… y realizar la navegación manualmente. De esta manera se irán añadiendo automáticamente todos los objetos con los que se interactúan al repositorio de objeto y todas las acciones que se realizan sobre los mismos. Este método es el más rápido si no existen problemas de ralentización con el software que se está automatizando.

- ✓ Añadir los objetos de las ventanas que se necesitarán para llevar a cabo el ciclo completo de grabación. Lo ideal es hacerlo sobre un repositorio de objetos global para poder reutilizarlo posteriormente (se tratará en el siguiente punto). Y luego introducir manualmente las acciones, comprobaciones, recogida de datos… que se pretende realizar sobre esos objetos. Esta manera de grabar requiere de un mayor conocimiento de la herramienta, pero también aporta un mayor control sobre las instrucciones introducidas.

Repositorio de objetos

El Repositorio de Objetos (Repository Object) es el lugar centralizado donde se encuentran almacenados los objetos de cualquier aplicación / página web, con sus propiedades y características.

Este repositorio es necesario para poder reconocer los objetos y que facilite la interactuación con ellos a través de UFT. Mientras se ejecuta el script, los objetos de la aplicación se identifican y reconocen por la comparación de las propiedades de los mismos con las propiedades almacenadas en el repositorio de objetos. De esta manera la ejecución de pasos de prueba es posible.

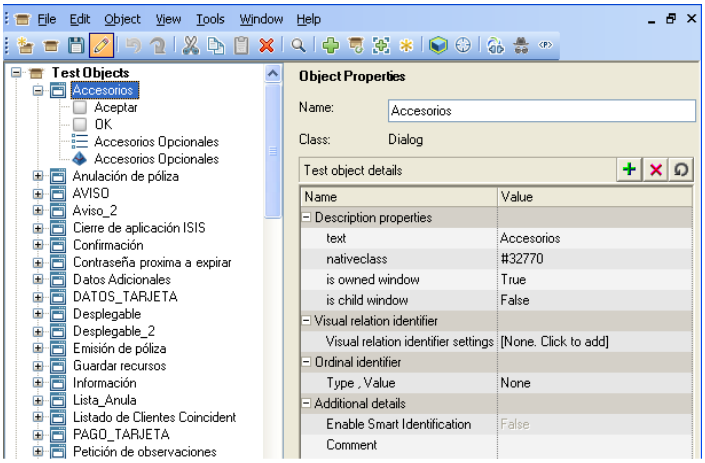
Existen dos tipos de repositorios de objetos:

- ✓ Repositorio global: este tipo de repositorio de objetos es la opción preferida cuando se trata con objetos dinámicos, por lo general llamado en múltiples pruebas o scripts.

- ✓ Repositorio local: es el repositorio de objetos por acción, y es la configuración predeterminada en el programa.

Desde UFT para abrir el repositorio de objetos se accederá a Resources > Object Repository… o bien a Resources > Object Repository Manager… según las necesidades requeridas para editar los repositorios, agregar o quitar objetos, modificar propiedades etc.

En esta ventana se refleja un árbol desplegable con los objetos almacenados y cuando se selecciona uno de ellos se listan las propiedades de tal objeto.



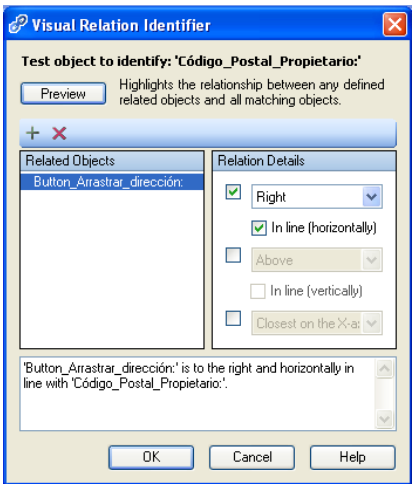
Propiedad de los objetos

La propiedades de los objetos son los valores identificativos y representativos de cada uno de los elementos que van a conformar nuestro script.

El reconocimiento de GUI en UFT es orientado a objetos. Para identificar un objeto UFT necesita una combinación única de propiedades del objeto.

También se puede identificar un objeto a través de su índice, localización etc.

Por último, se puede localizar el objeto por su posición espacial y relativa con respecto al resto de objetos dentro de la aplicación que se está automatizando. Esto se hace mediante Visual Relation Identifier.

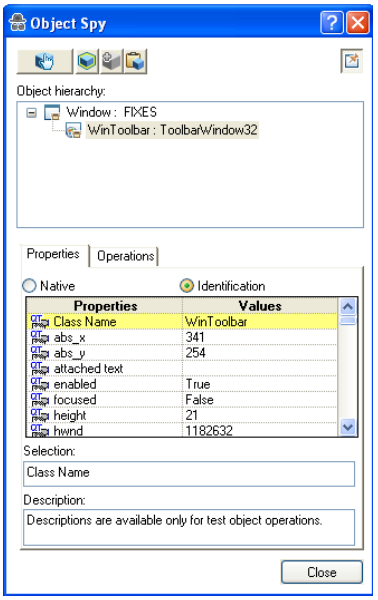


Espía de objetos

El espía de objetos (Object Spy) es una herramienta muy importante que contiene UFT, la cual permite conocer la estructura básica de un objeto de la prueba. Esta estructura es presentada en forma de árbol, reflejando las relaciones, objetos padre e hijos que contiene.

Además también se puede ver las propiedades del objeto de prueba y los métodos de cualquier objeto en la aplicación abierta.

Para abrir el Espía de Objetos se consigue a través de Tools > Object Spy…



➢ Funcionalidades de ALM

Organización de HP ALM

- ❑ Es una herramienta Web que gestiona todos las actividades básicas de un proceso de pruebas, incluyendo la gestión inicial de los requisitos y por último la gestión de los defectos.

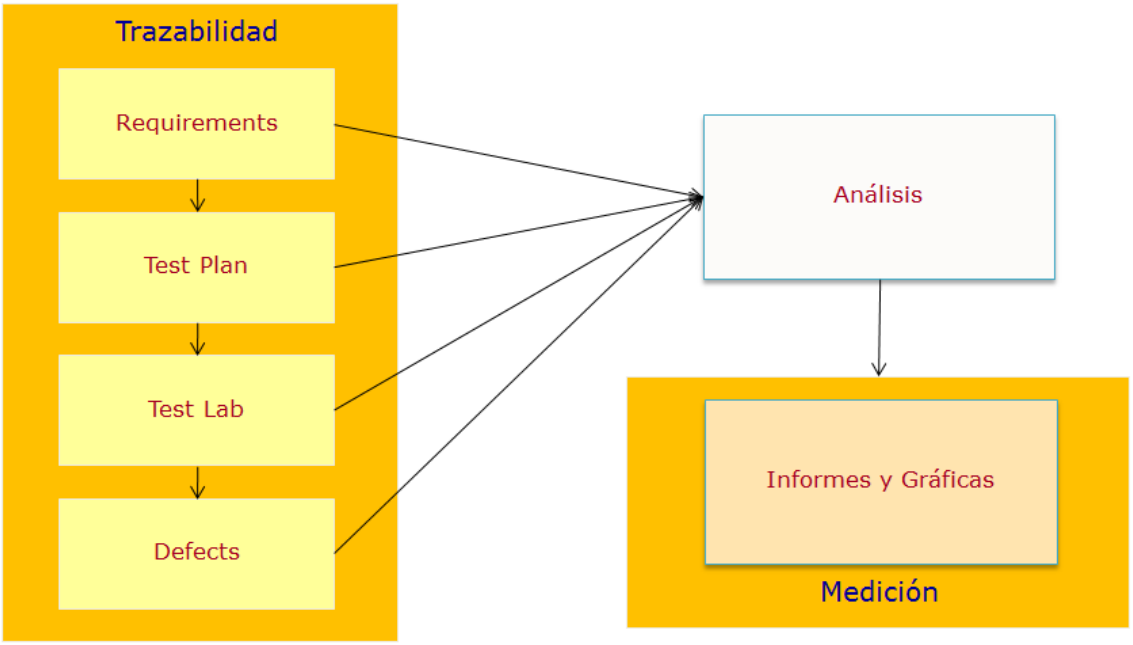
- ❑ Ofrece un entorno de trabajo organizado para evaluar la calidad de las aplicaciones de Software, agilizando el mantenimiento de un repositorio de datos completo, versionable, trazable y reutilizable.

- ❑ Dispone de APIs para integrar soluciones terceras o de la misma suite con el objetivo de conformar una plataforma única y global (HP Unified Functional Testing y HP LoadRunner).

- ❑ Facilita la importación y exportación de información mediante plugins Word y Excel.

- ❑ Proporciona mecanismos sencillos para analizar los resultados y generar informes / cuadros de mando establecidos por defecto y también personalizables.

Cada módulo permite gestionar un tipo de activo en particular. ALM permite la integración, la trazabilidad y la medición de la información.

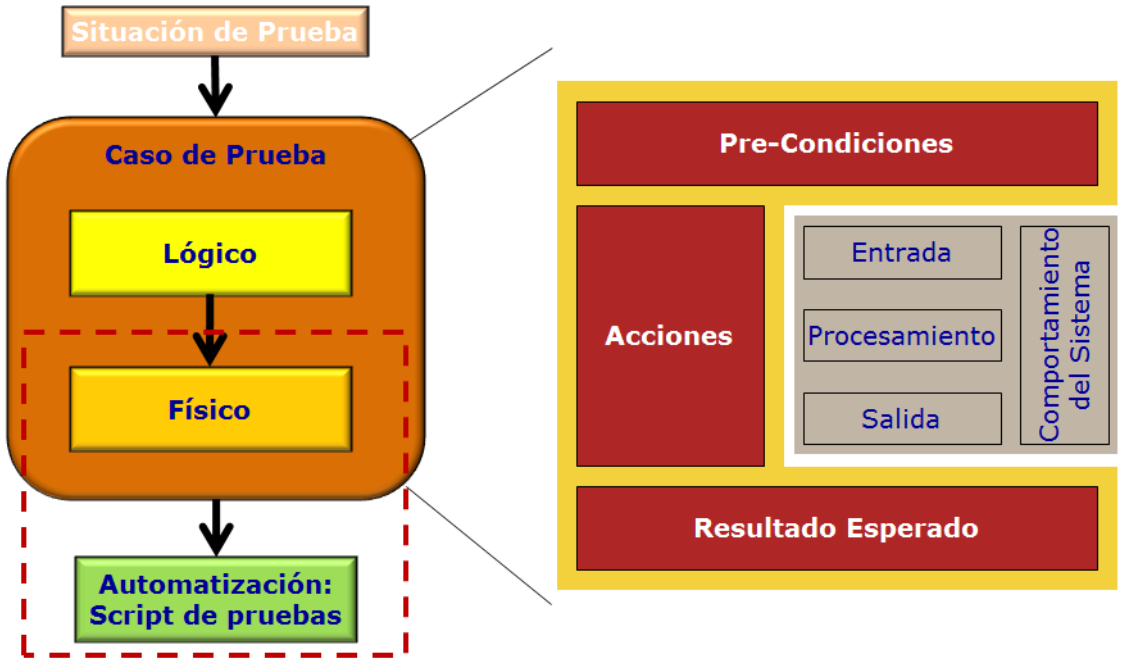


Casos de Prueba / Scripts de Prueba

Conceptos básicos:

- **Situación de Prueba**
  - ✓ Comportamiento aislado del objeto de prueba que se necesita probar.
- **Caso de Prueba Lógico**
  - ✓ Describe, en términos lógicos, el comportamiento a probar.
  - ✓ Cubre 1 o más situaciones de prueba.
- **Caso de Prueba Físico**
  - ✓ Cada caso lógico da lugar a un caso físico.
  - ✓ Se determinan todos los valores requeridos de:
    - ➢ Entradas
    - ➢ Configuración y estado del entorno y acciones
    - ➢ Resultados esperados
- **Script de Prueba**
  - ✓ Conjunto de instrucciones de código (agrupada en una rutina) cuya ejecución permite saber si el sistema bajo pruebas tiene el comportamiento funcional requerido.

Estructura:



Introducción/visión general contiene información general acerca de los Casos de Prueba.

- ✓ ***Identificador*** es un identificador único para futuras referencias, por ejemplo, mientras se describe un defecto encontrado.
- ✓ ***Creador*** es el nombre del analista o diseñador de pruebas, quien ha desarrollado pruebas o es responsable de su desarrollo.
- ✓ ***Versión*** la actual definición del caso de prueba.
- ✓ ***Nombre del caso de prueba*** debe ser un título entendible por personas, para la fácil comprensión del propósito del caso de prueba y su campo de aplicación.
- ✓ ***Identificador de requerimientos*** el cuál está incluido por el caso de prueba. También aquí puede ser identificador de casos de uso o especificación funcional.
- ✓ ***Propósito*** contiene una breve descripción del propósito de la prueba, y la funcionalidad que chequea.

Actividad de los casos de prueba

- ✓ ***Configuración Inicial / Dependencias*** contiene información acerca de la configuración del hardware o software en el cuál se ejecutará el caso de prueba., así como posibles dependencias con otros casos de prueba.
- ✓ ***Acciones pasos*** a realizar para completar la prueba, incluyendo la descripción de los datos de entrada.
- ✓ ***Configuración Final*** describe las acciones que deben ser ejecutadas después de ejecutado el caso de prueba.

Resultados

- ✓ ***Resultados esperados*** contiene una descripción de lo que el tester debería ver tras haber completado todos los pasos de la prueba
- ✓ ***Resultados reales*** contienen una breve descripción de lo que el tester encuentra después de que los pasos de prueba se hayan completado.

¿ Qué es un "buen" caso de prueba ?

Un caso de prueba (físico) consiste en:

- ***pre-condiciones*** de ejecución.
- conjunto de valores de **entrada**.
- resultados esperados.
- ***post-condiciones*** de ejecución.

Permite la detección efectiva de defectos.

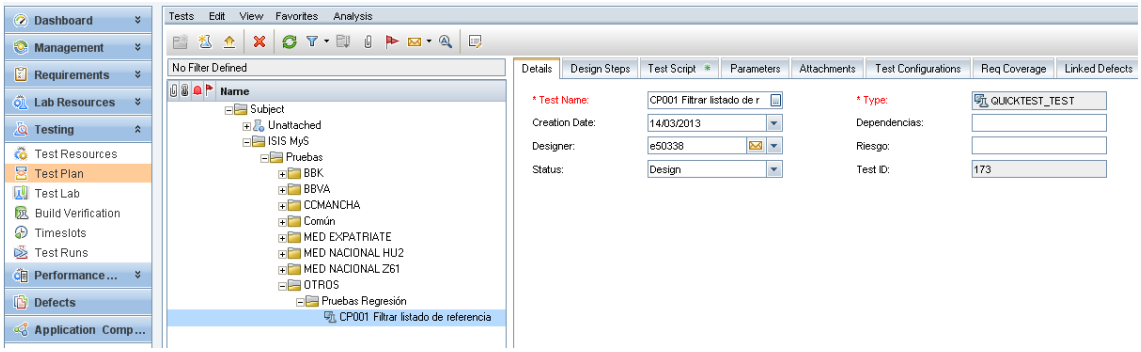
Es reproducible, transferible y mantenible.

Minimiza la dependencia entre la persona que lleva a cabo la especificación y aquella que lleva la ejecución.

Conforma la base para garantizar la Calidad del Software.

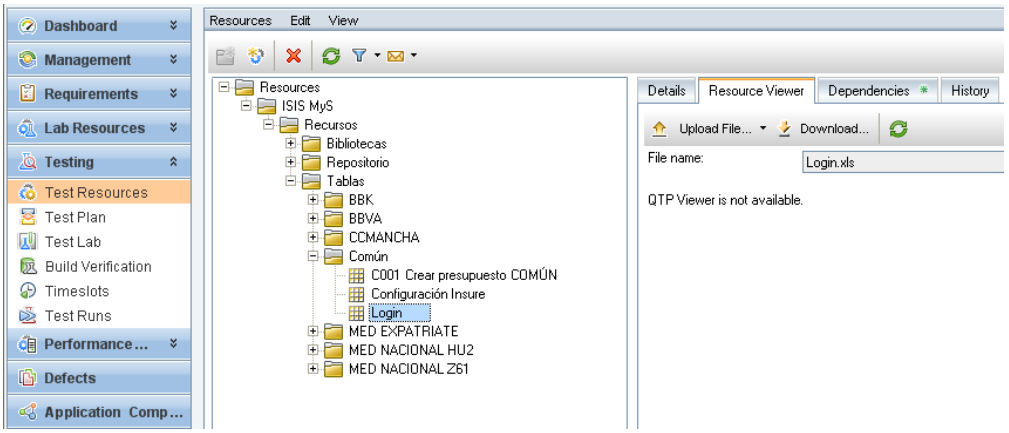
ALM - Test Plan

En este módulo es donde se gestiona la especificación de los casos de prueba / scripts de prueba.



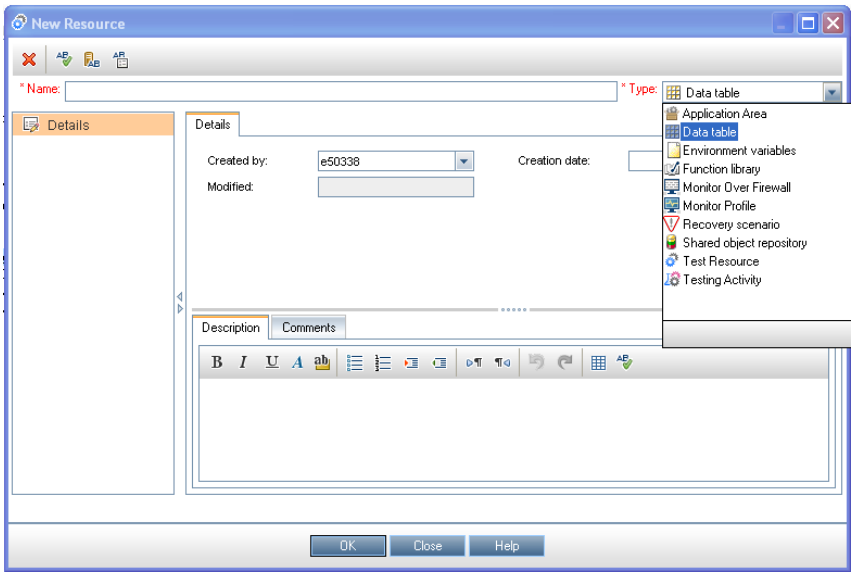
Recursos compartidos:

- Desde ALM se pueden utilizar recursos compartidos (Test Resources), los cuales podrán ser requeridos y compartidos, si así lo se desea, por los casos generados.



- Estos recursos compartidos pueden ser de varios tipos:

  - ✓ Tabla de datos (Data table)
  - ✓ Variables de entorno (Environment variables)
  - ✓ Biblioteca de funciones (Function library)
  - ✓ Shared object repository (Repositorio de objetos compartido)
  - ✓ …



  - ✓ Para utilizar estos recursos en los casos, se deberán de definir en ALM y posteriormente asociarlos a los mismos, para poder interactuar con ellos.

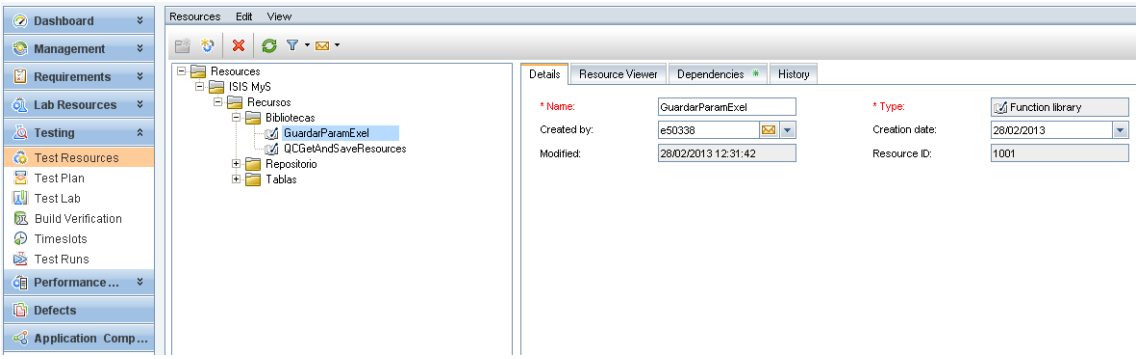
  - ✓ Toda la gestión de los recursos compartidos se materializará accediendo a través del menú de ALM por Testing > Test Resources.

Librerías de intercambio de datos:

- Se trata de un tipo de recurso compartido (Test Resources), el cual vendrá denominado en ALM como un recurso de tipo Biblioteca de funciones (Function library).

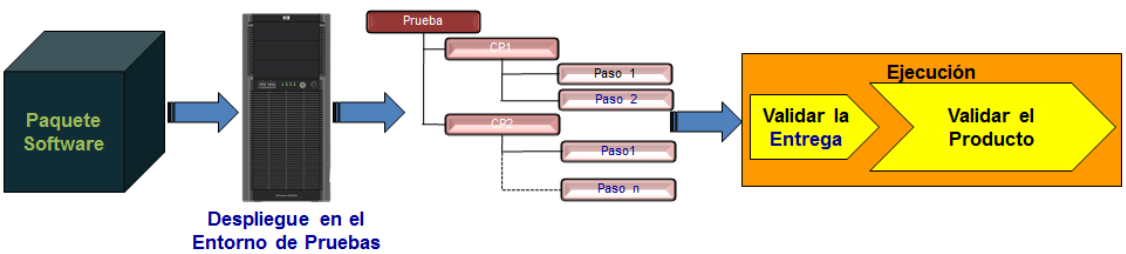
- Este tipo de recurso permite introducir librerías creadas y no definidas por defecto. Se utilizarán para poder utilizar las funciones definidas dentro de la misma y de esta manera poder realizar operativas más o menos complejas con una simple llamada.

- La principal ventaja de utilizar este tipo de librerías es que se unifica la metodología con el consiguiente ahorro al no tener que repetir código y minimizando la complejidad de los scripts a los que hace referencia los casos.



Laboratorio de Ejecución:

Se utiliza para validar el producto software



Los objetivos son:

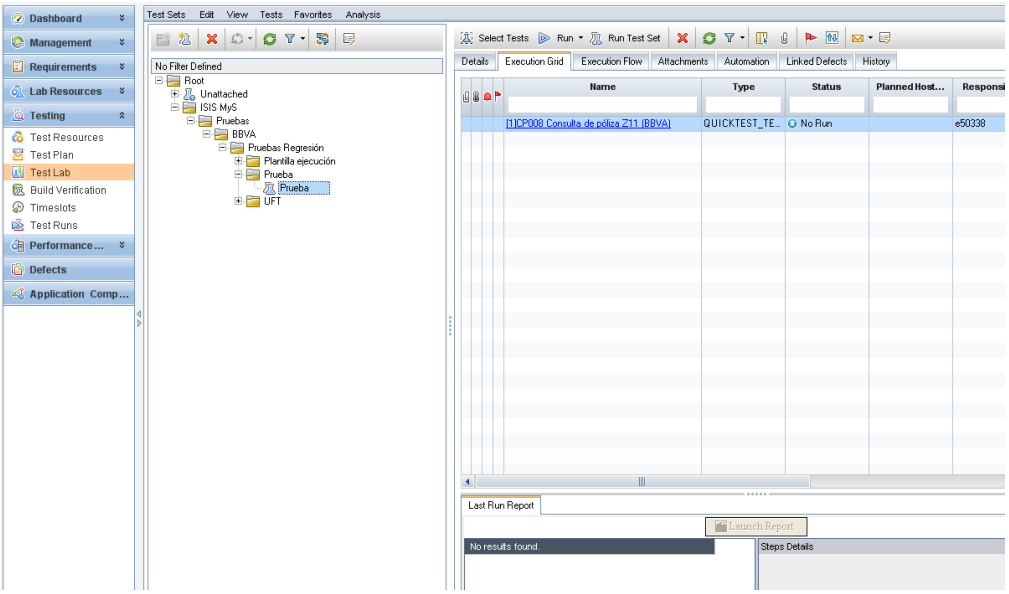
- Detectar, mediante la ejecución de los casos de prueba manuales, el máximo de defectos posibles sobre una versión específica del software.

- Asegurar, mediante la ejecución de los scripts de prueba de regresión, el correcto funcionamiento del producto tras la implementación de cambios, evolutivos o correctivos, en la versión del software.

Planificación de ciclos de ejecución

ALM - Test Lab

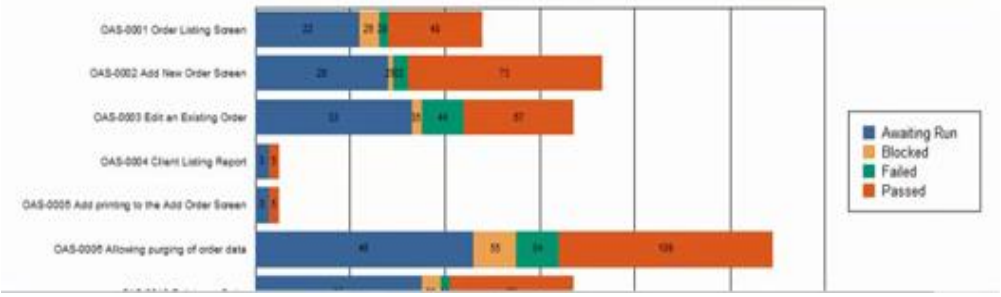
En este módulo es donde se gestionan los ciclos de ejecución de las pruebas.



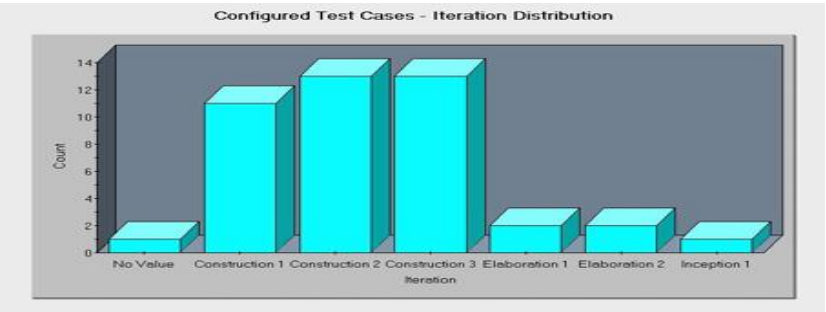
Análisis y Reporting:

Se trata de medir la **Calidad alcanzada** una vez terminada la ejecución de los ciclos de pruebas.

1. Métricas e Indicadores de la **Calidad del Producto**



2. Métricas e Indicadores de la **Productividad de la fase de Pruebas**



Ejemplos de Indicadores:

a. **Seguimiento de la corrección de los defectos:** Se trata de medir la progresión de las correcciones de los defectos, una vez detectados.
b. **Seguimiento de la cobertura de pruebas:**Se trata de medir, por cada uno de los atributos de Calidad establecido, el estado de los ciclos de pruebas y la cobertura de los requisitos alcanzada.
c. **Origen de los defectos detectados (Root Cause Analysis):** Se trata de medir el ratio de defectos detectados por el equipo de Pruebas y de analizar el origen de los mismos.
d. **Cumplimiento de los plazos planificados:** Se trata de medir el cumplimiento de las planificaciones previstas en las fases de Especificación y Ejecución.
e. **Volumen de casos de prueba diseñados / ejecutados:** Se trata de medir el número de casos de prueba diseñados / ejecutados frente a los planificados.
f. **Fiabilidad de los defectos reportados: S**e trata de medir la veracidad de los defectos detectados por el equipo de Pruebas. Para ello se calcula el ratio de defectos cancelados tras rechazos confirmados desde Desarrollo.
g. **Volumen de defectos:** Se trata de medir el número de defectos detectados por ciclos de pruebas. También se puede contrastar este número con el número de defectos que se detectan en niveles posteriores.

Cuadro de mando:

	MÉTRICA	MEDIDA (%)	UMBRAL REQUERIDO (%) Valor=10	UMBRAL SUFICIENTE (%) Valor=5	UMBRAL INSUFICIENTE (%) Valor=0	IND.Q Valores 10, 5, 0	PESO Q (%)	IND.M Valor entre 0 y 10	PESO M (%)	IND.G Valor del 1 al 5
COBERTURA	Cobertura de Elementos a Probar - Riesgo Crítico		100%	> 90%	<= 90%		50%		20%	
	Cobertura de Elementos a Probar - Riesgo Alto		> 90%	> 80%	<= 80%		30%			
	Cobertura de Elementos a Probar - Riesgo Medio		> 80%	> 60%	<= 60%		15%			
	Cobertura de Elementos a Probar - Riesgo Bajo		> 60%	> 50%	<= 50%		5%			
EJECUCIÓN	Ejecución de Casos de Prueba asociados a Elementos a Probar - Riesgo Crítico		> 90%	> 80%	<= 80%		50%		10%	
	Ejecución de Casos de Prueba asociados a Elementos a Probar - Riesgo Alto		> 80%	> 60%	<= 60%		30%			
	Ejecución de Casos de Prueba asociados a Elementos a Probar - Riesgo Medio		> 60%	> 50%	<= 50%		15%			
	Ejecución de Casos de Prueba asociados a Elementos a Probar - Riesgo Bajo		> 50%	> 40%	<= 40%		5%			
DEFECTOS BLOQU.	Defectos Abiertos Bloqueantes asociados a Elementos a Probar - Riesgo Crítico		0%	<= 5%	> 5%		50%		40%	
	Defectos Abiertos Bloqueantes asociados a Elementos a Probar - Riesgo Alto		<= 5%	<= 10%	> 10%		30%			
	Defectos Abiertos Bloqueantes asociados a Elementos a Probar - Riesgo Medio		<= 10%	<= 20%	> 20%		15%			
	Defectos Abiertos Bloqueantes asociados a Elementos a Probar - Riesgo Bajo		<= 20%	<= 30%	> 30%		5%			
DEFECTOS SEVEROS	Defectos Abiertos Severos asociados a Elementos a Probar - Riesgo Crítico		<= 5%	<= 10%	> 10%		50%		15%	
	Defectos Abiertos Severos asociados a Elementos a Probar - Riesgo Alto		<= 10%	<= 20%	> 20%		30%			
	Defectos Abiertos Severos asociados a Elementos a Probar - Riesgo Medio		<= 20%	<= 30%	> 30%		15%			
	Defectos Abiertos Severos asociados a Elementos a Probar - Riesgo Bajo		<= 30%	<= 40%	> 40%		5%			
DEFECTOS MEDIOS	Defectos Abiertos Medios asociados a Elementos a Probar - Riesgo Crítico		<= 10%	<= 20%	> 20%		50%		10%	
	Defectos Abiertos Medios asociados a Elementos a Probar - Riesgo Alto		<= 20%	<= 30%	> 30%		30%			
	Defectos Abiertos Medios asociados a Elementos a Probar - Riesgo Medio		<= 30%	<= 40%	> 40%		15%			
	Defectos Abiertos Medios asociados a Elementos a Probar - Riesgo Bajo		<= 40%	<= 50%	> 50%		5%			
DEFECTOS LEVES	Defectos Abiertos Leves asociados a Elementos a Probar - Riesgo Crítico		<= 10%	<= 20%	> 20%		50%		5%	
	Defectos Abiertos Leves asociados a Elementos a Probar - Riesgo Alto		<= 20%	<= 30%	> 30%		30%			
	Defectos Abiertos Leves asociados a Elementos a Probar - Riesgo Medio		<= 30%	<= 40%	> 40%		15%			
	Defectos Abiertos Leves asociados a Elementos a Probar - Riesgo Bajo		<= 40%	<= 50%	> 50%		5%			

1	2	3	4	5

Se establecen cinco valores y colores para indicar la calidad funcional del proyecto:
- 5 = No cumple las expectativas requeridas - Estado intolerable
- 4 = No cumple aun las expectativas requeridas - Estado próximo al tolerable
- 3 = Estado tolerable
- 2 = Cumple las expectativas - Estado próximo al requerido
- 1 = Cumple al máximo las expectativas - Estado requerido

➢ Gestión de Defectos

Un defecto de software, es el resultado de encontrar un fallo o deficiencia en el producto de software con respecto a sus especificaciones (funcionales y no funcionales).

Dicho fallo puede presentarse en cualquiera de las etapas del ciclo de vida del software aunque los más evidentes se dan en la etapa de Construcción.

Tipos de defectos más usuales:

- ✓ Defectos de Software: Fallo con respecto a las especificaciones funcionales.
- ✓ Defectos de Pruebas: El propio caso o script de prueba contiene algún paso erróneo.
- ✓ Defectos de Entorno: Una pieza del entorno de entorno de prueba está desplegada o configurada de manera incorrecta.
- ✓ Defectos de Datos: Los datos son inconsistentes.

Atributos más usuales

En la composición de la descripción de un defecto, se recomienda incluir una serie de atributos cuya información detallada permita su reproducción.

- ✓ *Identificador:* Es un identificador único para el defecto.
- ✓ *Título:* Breve descripción del defecto.
- ✓ *Descripción: Descripción lo más detallada y especifica posible de las acciones llevadas a cabo sin omitir detalle alguno. En la misma se deben incluir los pasos para su reproducción y los datos utilizados.*
- ✓ *Fecha detección: Del defecto.*
- ✓ *Fecha de resolución: A cumplimentar por desarrollo.*
- ✓ *Estado: En su ciclo de vida.*
- ✓ *Persona y/o grupo de detección: Persona o grupo que ha identificado el defecto.*
- ✓ *Versión de detección: Versión de software en la que se ha detectado el defecto.*
- ✓ *Severidad: Grado de criticidad del defecto. (Alta, media y baja).*
- ✓ *Prioridad: Grado de priorización respecto a objetivos de ejecución, funcionales o de proyecto.*
- ✓ *Estado: Estado en el que se encuentra el defecto.*
- ✓ *Persona y/o grupo asignado: Identificador de la persona y/o grupo que tiene asignado el defecto en el momento y debe aportar una respuesta.*

- ✓ *Comentarios: Información adicional que se considere necesaria para la resolución del defecto.*

- ✓ *Historial: Listado de resoluciones, generadas habitualmente de forma automática por la herramienta de gestión.*

Ciclo de vida "estándar" – Transición de estados

El estado de un defecto determina en qué fase de elaboración se encuentra. Las posibles caracterizaciones pueden ser:

- ✓ ***Nuevo*** *– el error, defecto o incidencia ha sido notificado por primera vez por parte del tester.*

- ✓ **Abierto** – la notificación se liberó por parte del gestor de pruebas.

- ✓ **Rechazado** – la notificación fue rechazada por el gestor de pruebas.

- ✓ **Investigado** – el desarrollador intenta identificar el error.

- ✓ **Observación** – no se puede identificar el error, está en observación.

- ✓ **En proceso** – el error se ha liberado para su corrección (en caso contrario se rechaza).

- ✓ **Repetición de la prueba** –se asigna por parte del desarrollador después de la corrección.

- ✓ **Solucionado** – se asigna por el tester cuando el error se demuestra eliminado después de la repetición de la prueba.

- ✓ **No solucionado** – la asigna el tester cuando el error continúa apareciendo.

Ciclo de vida "estándar" – Actores:

- **Área de Testing** – recurso que a través de sus pruebas idéntica, encuentra el defecto. Una vez resuelto, procede con la verificación de la corrección realizada y el cierre o reapertura del defecto si procede.

- **Desarrollo. "Gestor de Defectos"** – responsable dentro de desarrollo de confirmar la valía. reproducibilidad del defecto reportado. Una vez resuelto, poner a disposición del equipo de Testing para su verificación y cierre.

- **Desarrollo. "Gestión de la configuración"** – responsable dentro de desarrollo de confirmar, verificar la correcta versión así como del despliegue del producto para las pruebas.

- **Desarrollo. Desarrollador** – recurso encargado de validar el defecto y proceder con su corrección.

- **Responsable de Entornos** – verificar el defecto reportado y proceder con la corrección necesaria. Disponibilidad del entorno de pruebas, datos de pruebas.

- **Jefe de Proyecto** – responsable de la gestión de defectos en la documentación suministrada. Defectos asumidos, pospuestos, ciclos de regresión.

Seguimiento: Sin un proceso de gestión de defectos en funcionamiento no se puede tener un tratamiento ordenado de los mismos.

El proceso de gestión de defectos debe construirse al principio del proyecto.

Algunos puntos que deben ser considerados para una gestión correcta de defectos son:

- Proporcionar notificaciones / formularios de notificación comunes.

- Todos las defectos / incidencias / errores / problemas deben ser recogidos, independientemente de parte de qué persona involucrada en el proceso provengan y de dónde hayan aparecido.

- Se debe definir una manera de proceder determinada para el tratamiento de los defectos.

En la gestión de incidencias se recogen todos las incidencias / errores encontrados en el proyecto.

Además de su recogida se actualiza constantemente los atributos del defecto hasta su cierre:

- Se determina la gravedad del error.

- Estado del error - p.ej., abierto, en proceso de trabajo, solucionado, etc.

Toda la información se almacena en una base de datos central:

- Se proporciona una visión óptima y única acerca de los errores encontrados y su tratamiento.

- Principal fuente de alimentación para los informes de seguimiento y cuadros de Mando.

➢ Estrategia y Preparación

Flujos principales:

Antes de comenzar a realizar la grabación de los nuevos casos se debe de realizar una lectura comprensiva de los pasos a seguir en cada uno de ellos y realizar una navegación manual para comprobar que tanto las funcionalidades, como los datos de entrada utilizados, los resultados obtenidos… son los esperados y los que se indican a través del plan de pruebas.

Tras comprobarse que los ciclos tienen una descripción correcta se debe de localizar los flujos principales de dichos casos. Es decir, el flujo más importante dentro del caso y generalmente el punto de entrada al mismo.

Si más de un caso contienen el mismo flujo principal es de gran ayuda el localizarlo y grabarlo como un Action de tipo reusable. De esta manera se podrá reutilizar una misma operativa entre diferentes casos.

Tras la reutilización de flujos principales se consigue:

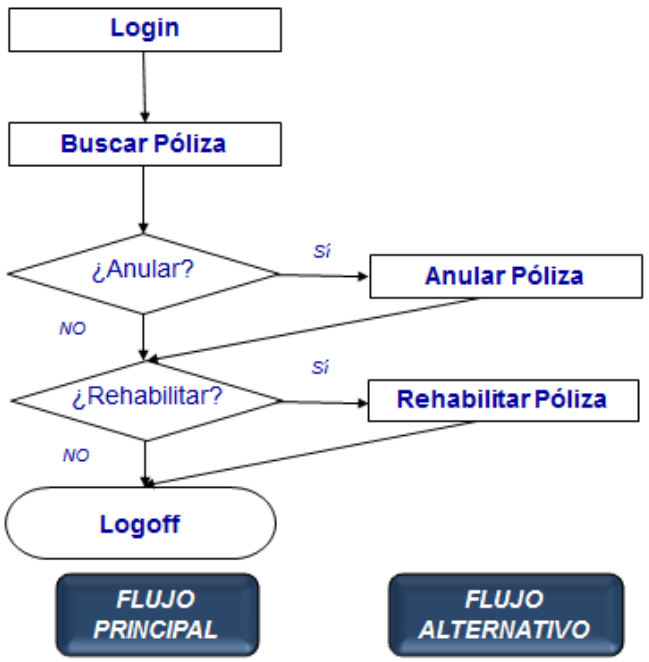
- ✓ Ahorro de tiempos
- ✓ Mayor eficiencia
- ✓ Ahorro de espacio de almacenamiento
- ✓ …

Flujos alternativos:

Son aquellos subflujos alternos que dependen del flujo principal y que realizan diferentes operativas adicionales a las del bloque básico.

Estos flujos suelen se Action locales de cada caso, debido a que su funcionalidad no se repite para el resto de los casos.

Los Action principales y los alternativos pueden ir alternándose durante el ciclo de prueba completo.



Integración ALM

ALM actúa como el elemento principal para las integraciones de las herramientas de HP. Trabaja como un "repositorio central", ALM guardará toda la información de las herramientas, tanto de pruebas automatizadas, como monitorizaciones etc., pudiendo interactuar con ellas y realizar cambios en las mismas.

Dadas las capacidades de ALM para ese tipo de tareas, el trabajo es más sencillo y los equipos de trabajo podrán conectarse de una manera sencilla y muy intuitiva. De esta manera se eliminan las "barreras" que hacen que un proyecto se ralentice y se trabaje con lentitud. Este tipo de integraciones es muy común y altamente recomendable para proyectos Agile, ya que la interactividad es muy alta y rápida.

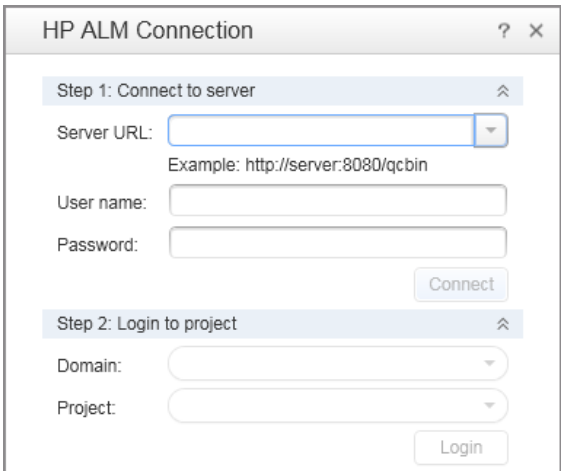
UFT se integra perfectamente con Quality Center para convertir pruebas manuales en automáticas y así ahorrar tiempo en pruebas y en pasos repetitivos.

La integración de UFT se realiza con un Add-in. Este, realizará una conexión entre los dos productos y permitirá expandir el campo en el que se está trabajando.

Se puede realizar la integración de un script de UFT de dos maneras diferentes:

- ✓ Realizando un test manual en ALM e introduciéndolo en una batería de pruebas mediante el módulo de "Test Plan", donde aparecerá un botón con el logotipo o el icono de UFT. En el momento que se pulsa el botón el software redirige automáticamente a UFT para poder comenzar a automatizar y realizar el código correspondiente.

- ✓ Desde UFT se puede conectar a ALM, accediendo a ALM > ALM Connection…, donde se debe realizar el login en el servidor y proyecto de ALM deseado. Entonces se crea un nuevo test seleccionando la ruta de ALM donde se desea que se aloje dicho nuevo caso. A partir de aquí todos los cambios que se guardan desde UFT se guardarán directamente en ALM, ya que es donde se encuentra ubicado el caso.



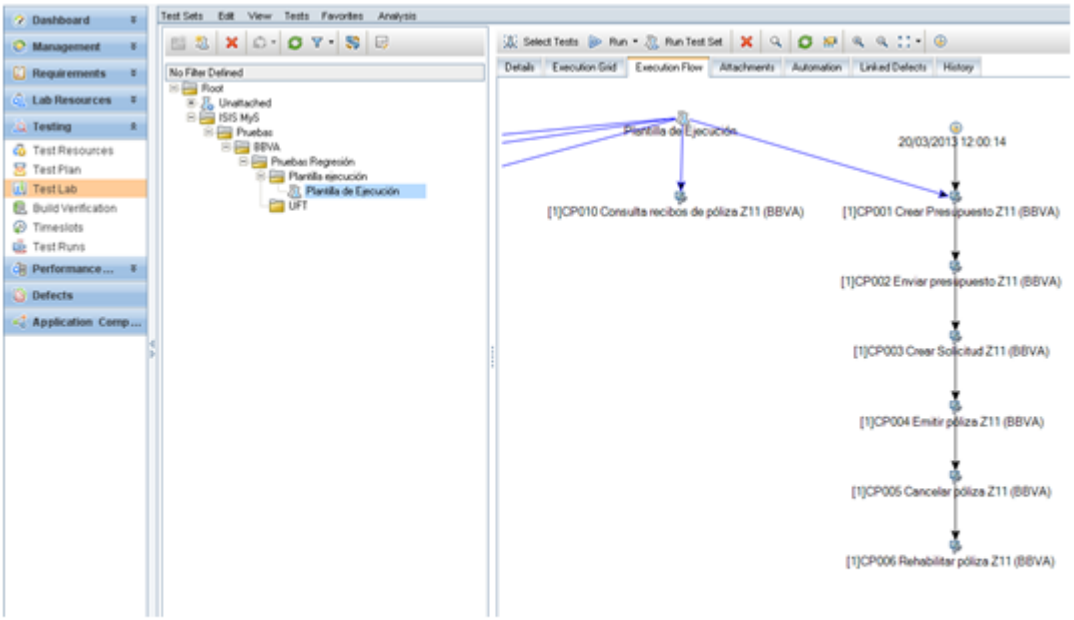
Preparación de árboles de ejecución:

Desde ALM se puede instaurar un árbol de ejecución (Execution Flow) en el que se podrá definir las dependencias entre los casos creados, el orden de ejecución entre ellos, la hora de ejecución, tiempos entre casos etc.

Para acceder al árbol de ejecución se tendrá que acceder al Test Lab y tener al menos un Test Set creado, el cual contendrá una parrilla de lanzamiento con una serie de casos previamente añadidos.

En nuestro Test Set se seleccionan la pestaña Execution Flow y ahí aparecerá el árbol de ejecución. Los objetos más representativos del mismo son:

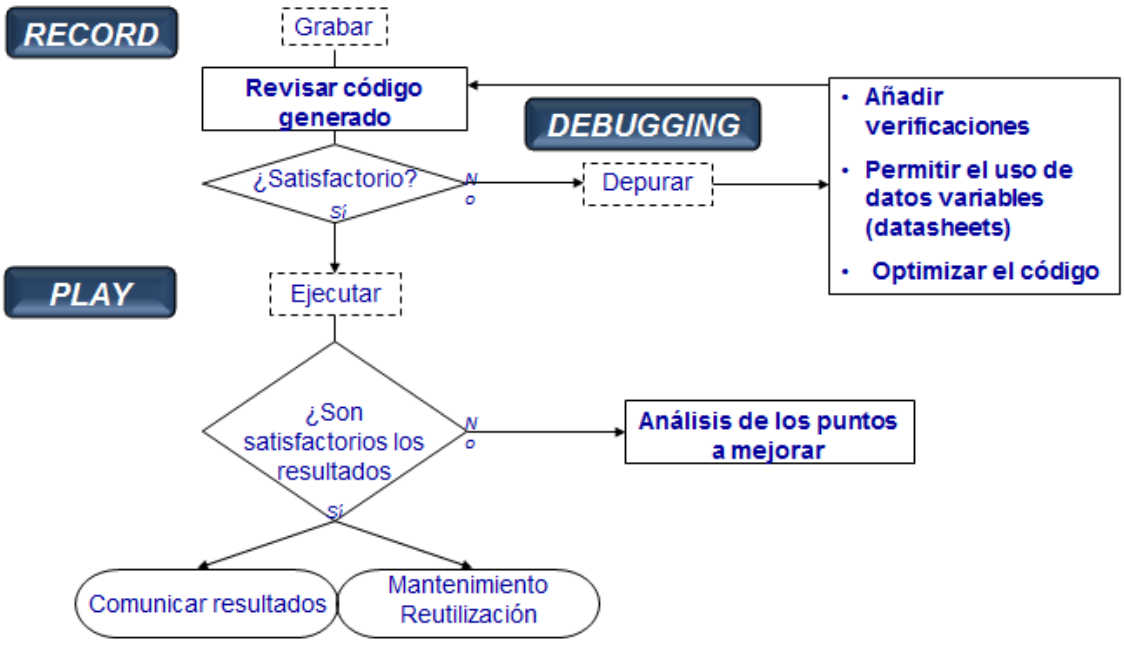
- ✓ Test Set principal
- ✓ Casos de prueba
- ✓ Dependencias (se representan mediante flechas)
- ✓ Tiempos (Time dependency)
- ✓ …



Test Lab **Test Set** **Execution Flow**

➢ Flujo de Trabajo HP UFT + ALM

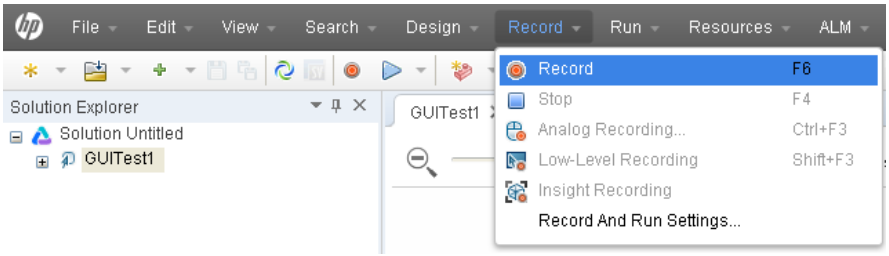
Grabación, Ejecución y Depuración:



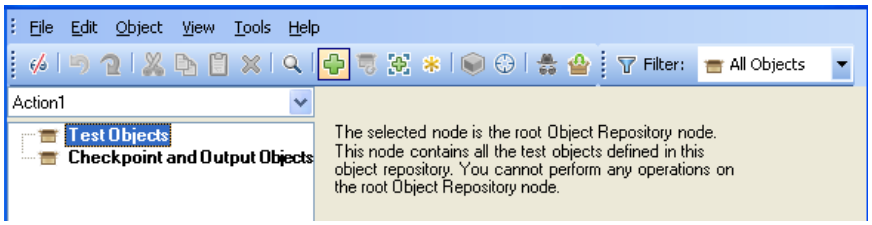
Grabación:

- Se abre UFT y se crea un nuevo test, añadiendo Add-in en caso de necesitarlo y de tipo GUI Test.
- Se realiza la grabación del ciclo, por cualquiera de los dos métodos de grabación expuestos en el apartado **Modos de grabación.**

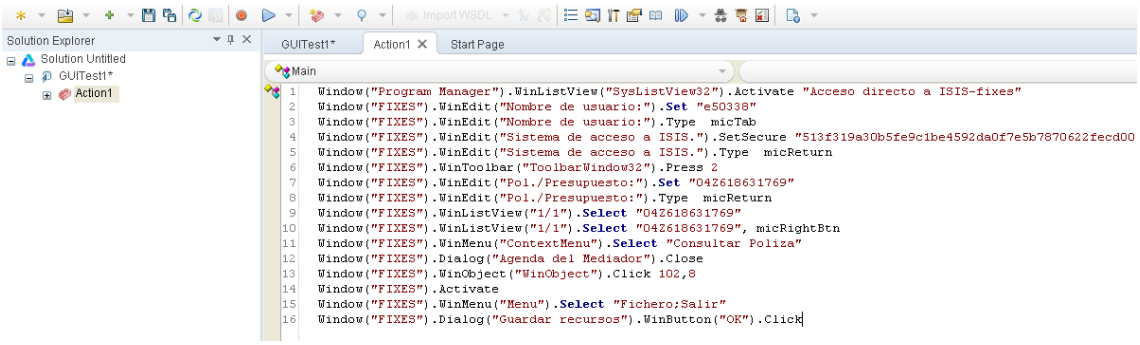
    - Grabación Directa



    - Grabación de los objetos que se necesitan en el repositorio y programación de las instrucciones necesarias.



El resultado es un script con instrucciones de la siguiente forma:



Depuración:

Ahora llega el momento de depurar el script. Esto se puede hacer realizando varias modificaciones sobre el script generado en la grabación. Algunas de estas modificaciones, que provocarán el tener un script más depurado y sólido son:

- Comprobaciones o checks de objetos
- Sincronismos y tiempos de espera
- Reutilización de Actions
- Utilización de métodos de intercambio de datos:
  - ✓ Parámetros
  - ✓ Tablas de datos
  - ✓ Ficheros
  - ✓ …
- Centralización de objetos en un repositorio global
- Reporte de eventos
- …

El resultado es un script más robusto y estable:

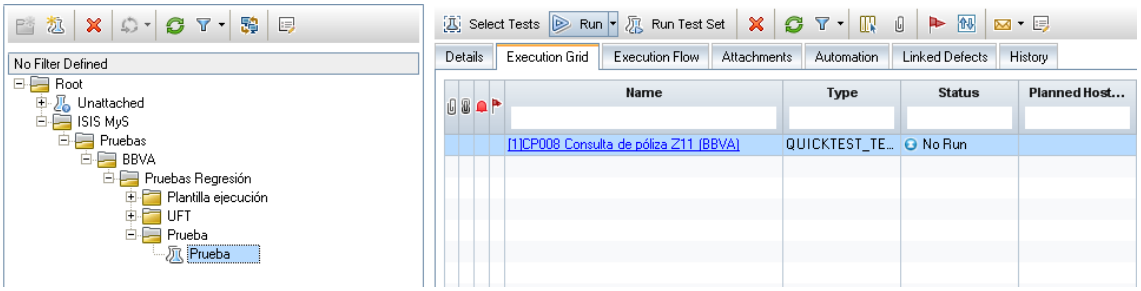
Ejecución:

Se realizará la ejecución del script elaborado para comprobar que funciona correctamente y se efectúa la navegación deseada desde el principio hasta el final (End to End).

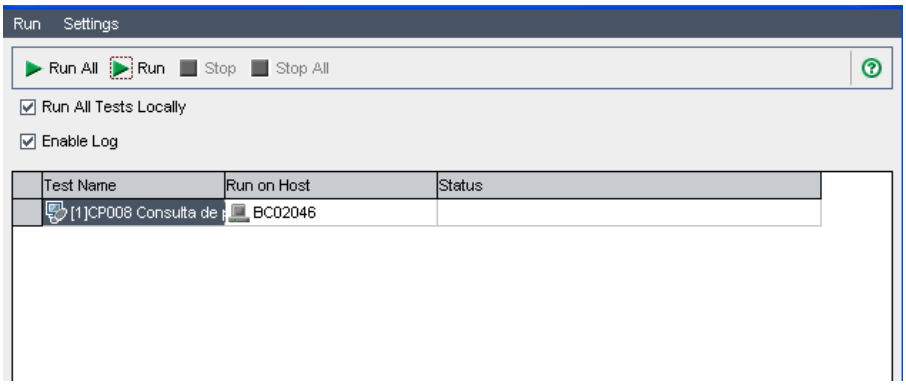
La ejecución desde UFT se hace accediendo a Run > Run…

Si diera error en algún punto, se tendría que ver lo que falla y depurar dicho problema para volver a ejecutarlo y que no se vuelvan a reproducir dichos problemas.

Si se desea ejecutarlo desde ALM, se debe guardar el caso en ALM en la ruta deseada. Desde ALM, dentro del Test Lab se crea un Test Set en cuya parrilla de lanzamiento se introduce dicho caso. Desde aquí se puede abrir seleccionando la fila del caso deseado y pulsando el botón “Run”.



Aparece la ventana desde donde se pulsa “Run”.



Cuando la ejecución ha finalizado, mostrará en “Status” si ha pasado o no correctamente.

Si se desea se puede ver también el informe de resultados.

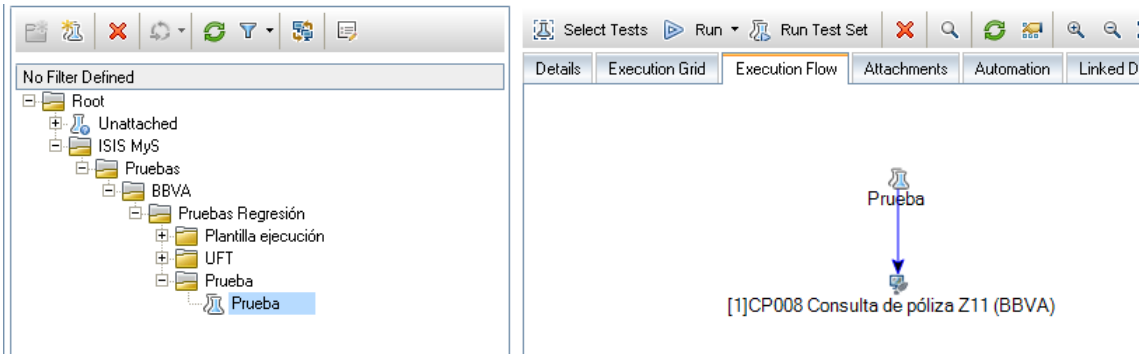
Integración en el árbol de ejecución:

Al acceder al árbol de ejecución todos los casos aparecen como independientes entre sí, ya que solo dependen del Test Set principal.

Si se desea que exista dependencia entre algunos de los casos, simplemente se tendrá que definir en el árbol mediante flechas. Siendo el

inicio de la flecha el caso que se ejecutará primero, y la punta de la flecha el caso que depende de él. Este caso dependiente no se ejecutará hasta que el anterior se haya ejecutado y/o haya finalizado correctamente (según la condición de ejecución que se haya definido previamente). Provocando el mismo efecto en posibles casos, que a su vez sean dependientes de él.

También se añadirá Time Dependency en el árbol si fuera requerido.



Anexo B

Encuestas realizadas.

Este anexo muestra la transcripción de los cuestionarios realizados a un grupo de 10 profesionales estrechamente ligados al ámbito del testing y de la calidad de software.

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: I.T. INFORMÁTICA DE SISTEMAS

Puesto técnico que ocupa: RESPONSABLE SERVICIO DE PRUEBAS

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	3	2	4	2
Portabilidad	5	5	2	1
Estética	4	5	2	2
Facilidad de uso	3	3	4	1
Seguridad	5	5	3	3
Mantenibilidad de los test	5	5	2	1
Ayuda y soporte	5	5	3	2
¿Cumple sus objetivos? (SI ó NO)	SI	SI	NO	NO

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: TÉCNICO SUP. DESARROLLO DE APLICACIONES INFORMÁTICAS

Puesto técnico que ocupa: INGENIERO DE PRUEBAS

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	3	3	2	3
Portabilidad	3	3	4	3
Estética	4	4	1	2
Facilidad de uso	5	4	3	3
Seguridad	3	3	3	3
Mantenibilidad de los test	4	4	2	3
Ayuda y soporte	3	3	1	2
¿Cumple sus objetivos? (SI ó NO)	SI	SI	NO	SI

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: LICENCIADO EN CIENCIAS MATEMÁTICAS (ESPECIALIDAD CIENCAS DE LA COMPUTACIÓN)

Puesto técnico que ocupa: INGENIERO SENIOR DE PRUEBAS

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	4	4	-	3
Portabilidad	3	4	-	2
Estética	4	4	-	3
Facilidad de uso	4	5	-	2
Seguridad	4	3	-	3
Mantenibilidad de los test	4	4	-	3
Ayuda y soporte	4	4	-	2
¿Cumple sus objetivos? (SI ó NO)	SI	SI	-	NO

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: GRADO SUPERIOR EN DESARROLLO DE APLICACIONES INFORMATICAS (DAI)

Puesto técnico que ocupa: INGENIERO DE PRUEBAS

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	3	3	3	2
Portabilidad	4	4	3	3
Estética	5	5	5	4
Facilidad de uso	4	4	4	3
Seguridad	5	5	5	5
Mantenibilidad de los test	4	4	4	4
Ayuda y soporte	5	5	3	3
¿Cumple sus objetivos? (SI ó NO)	SI	SI	SI	SI

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: INGENIERÍA SUPERIOR INFORMÁTICA

Puesto técnico que ocupa: INGENIERO DE PRUEBAS

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	4	4	2	2
Portabilidad	3	3	3	3
Estética	5	5	1	2
Facilidad de uso	3	3	3	2
Seguridad	4	4	3	3
Mantenibilidad de los test	4	4	2	2
Ayuda y soporte	4	4	2	2
¿Cumple sus objetivos? (SI ó NO)	SI	SI	NO	NO

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: INGENIERÍA SUPERIOR INFORMÁTICA

Puesto técnico que ocupa: SENIOR QA TESTER

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	4	4	3	3
Portabilidad	4	4	2	4
Estética	4	3	2	2
Facilidad de uso	4	3	2	3
Seguridad	3	4	3	3
Mantenibilidad de los test	4	4	2	3
Ayuda y soporte	4	4	3	3
¿Cumple sus objetivos? (SI ó NO)	SI	SI	NO	SI

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: INGENIERÍA TÉCNICA INFORMÁTICA

Puesto técnico que ocupa: ANALISTA Y DESARROLLADOR

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	5	3	2	1
Portabilidad	4	4	4	5
Estética	5	4	3	1
Facilidad de uso	5	4	5	1
Seguridad	4	4	2	2
Mantenibilidad de los test	4	4	2	1
Ayuda y soporte	3	3	3	1
¿Cumple sus objetivos? (SI ó NO)	SI	SI	NO	SI

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: INGENIERÍA TÉCNICA INFORMÁTICA DE SISTEMAS

Puesto técnico que ocupa: INGENIERO DE PRUEBAS

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	4	3	4	3
Portabilidad	3	4	3	3
Estética	5	4	3	3
Facilidad de uso	4	4	4	4
Seguridad	4	4	4	3
Mantenibilidad de los test	4	3	4	4
Ayuda y soporte	4	4	4	4
¿Cumple sus objetivos? (SI ó NO)	SI	SI	NO	NO

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: INGENIERÍA INFORMÁTICA

Puesto técnico que ocupa: TÉCNICO DE PRUEBAS

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	3	3	2	3
Portabilidad	3	3	3	4
Estética	4	4	2	2
Facilidad de uso	4	4	4	4
Seguridad	4	4	2	3
Mantenibilidad de los test	4	3	3	3
Ayuda y soporte	3	3	2	2
¿Cumple sus objetivos? (SI ó NO)	SI	SI	SI	SI

Encuesta de satisfacción aspectos de las herramientas

Dar un valor del 1 al 5 como resultado. Con este número se pretende concretar el nivel de satisfacción personal mediante la siguiente regla:

Pobre	Regular	Bueno	Muy Bueno	Excelente
1	2	3	4	5

Tras ello se realiza una pregunta directa acerca de si la herramienta satisface sus necesidades como usuario, y que se contestará con SI ó NO.

Formación: INGENIERÍA SUPERIOR INFORMÁTICA

Puesto técnico que ocupa: INGENIERO DE PRUEBAS

Capacidades de la herramienta	HP UFT	HP LoadRunner	Selenium IDE	Apache JMeter
Formación	4	2	4	2
Portabilidad	2	1	5	4
Estética	4	5	2	2
Facilidad de uso	4	4	3	1
Seguridad	4	2	1	1
Mantenibilidad de los test	4	5	3	1
Ayuda y soporte	5	4	3	1
¿Cumple sus objetivos? (SI ó NO)	SI	SI	SI	NO

Bibliografía

[1] Software Testing | SSTQB.
http://www.sstqb.es/

[2] Blog Historia de la Informática. Universidad Politécnica de Valencia. Diciembre 2010.
http://histinf.blogs.upv.es/2010/12/28/ingenieria-del-software/

[3] Dr. Juan Manuel Fernández Peña. Calidad de Software - Universidad Veracruzana. 2011.
http://www.uv.mx/personal/jfernandez/files/2010/07/8_Calidad.pdf

[4] I Jornada sobre Calidad del Producto Software e ISO 25000, Santiago de Compostela. Javier Garzás Parra, Fernando Suárez Lorenzo. Junio 2014.
http://www.cpeig.org/portal/system/files/175/2134/Libro+Jornadas+Galicia+Calidad+Software.pdf

[5] CES – CENTRO DE ENSAYOS DE SOFTWARE. Sección TESTING.
http://www.ces.com.uy/index.php/servicios/testing-independiente

[6] QA: Pruebas para asegurar la calidad del producto software (I). Elevenpaths blog offers. Septiembre 2014
http://blog.elevenpaths.com/2014/09/qa-pruebas-para-asegurar-la-calidad-del.html

[7] QA: Pruebas para asegurar la calidad del producto software (II). Elevenpaths blog offers. Noviembre 2014
http://blog.elevenpaths.com/2014/11/qa-pruebas-para-asegurar-la-calidad-del.html

[8] Ingeniería de Software: TIPOS DE PRUEBAS. Abril 2005
http://ing-sw.blogspot.com.es/2005/04/tipos-de-pruebas-de-software.html

[9] ¿Qué es el software libre? - Proyecto GNU - Free Software Foundation.
http://www.gnu.org/philosophy/free-sw.es.html

[10] Free Software Foundation.
https://www.fsf.org/es

[11] Diferencia entre Software Libre y Software Gratuito. Abril 2010
http://planetared.com/2010/04/diferencia-entre-software-libre-y-software-gratuito/

[12] Ventajas y desventajas del Software Propietario y del Software Libre. Abril 2010
http://tecnologiaedu.us.es/nweb/cursos/asig-nntt/html/karen-slu/3.htm#

[13] Software libre y software propietario. María Isabel Atopo. Julio 2011
http://www.monografias.com/trabajos89/sotware-libre-y-propietario/sotware-libre-y-propietario.shtml

[14] Open Source Initiative.
http://www.opensource.org

[15] ¿Por qué invertir en la automatización de pruebas Software? - Rational XDE Tester. Ana López-Mancisidor Rueda.
http://www.ati.es/IMG/pdf/IBM.pdf

[16] Estrategias existentes de automatización de pruebas funcionales y el Scriptless. Javier Garzás. Septiembre 2013.
http://www.javiergarzas.com/2013/09/automatizacion-de-pruebas-funcionales-1.html

[17] Una Introducción a las Pruebas de Prestaciones (V.1.2). José María Morales Vázquez. Septiembre 2013.
http://pics.unlugarenelmundo.es/hechoencasa/una%20introducci%C3%B3n%20a%20las%20pruebas%20de%20prestaciones.pdf

[18] Apache JMeter - Apache JMeter™. Installation Guide, System Requirements and User Guide.
http://jmeter.apache.org/

[19] HP UFT. Installation Guide, System Requirements and User Guide.
http://www8.hp.com/es/es/software-solutions/unified-functional-automated-testing/

[20] Selenium - Web Browser Automation. Installation Guide, System Requirements and User Guide.
http://www.seleniumhq.org/

[21] HP Loadrunner. Installation Guide, System Requirements and User Guide.
http://www8.hp.com/es/es/software-solutions/loadrunner-load-testing/

[22] TestComplete Platform
http://smartbear.com/product/testcomplete/overview/

[23] Ranorex
http://www.ranorex.com/

[24] Testopia - Mozilla
https://developer.mozilla.org/es/docs/Mozilla/Bugzilla/Testopia

[25] FWPTT web load testing framework
http://fwptt.sourceforge.net/

[26] Watir.com | Web Application Testing in Ruby
http://watir.com/

[27] IBM - Rational Functional Tester
http://www-03.ibm.com/software/products/es/functional

[28] Diagrama de Gantt Online Tom's Planner
http://www.tomsplanner.es/